

SUZAN KELLY BORGES

**RESOLUÇÃO DE TIMETABLING UTILIZANDO ALGORITMOS  
GENÉTICOS E EVOLUÇÃO COOPERATIVA**

Dissertação apresentada ao Curso de  
Mestrado em Informática, do Departamento  
de Informática, Setor de Ciências Exatas da  
Universidade Federal do Paraná.

Orientadora: Prof.<sup>a</sup> Dr.<sup>a</sup> Aurora T. Ramirez  
Pozo

CURITIBA  
2003




Ministério da Educação  
Universidade Federal do Paraná  
Mestrado em Informática

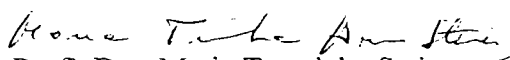
## PARECER

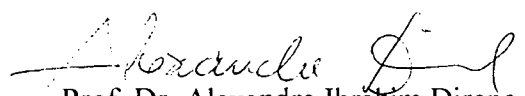
Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática, da aluna *Suzan Kelly Borges*, avaliamos o trabalho intitulado, “*Resolução de Timetabling Utilizando Algoritmos Genéticos e Evolução Cooperativa*”, cuja defesa foi realizada no dia 14 de outubro de 2003, às quatorze horas, no Auditório da Informática da Universidade Federal do Paraná. Após a avaliação, decidimos pela aprovação da candidata.

Curitiba, 14 de outubro de 2003.

  
Prof.ª Dra. Aurora Trinidad Ramirez Pozo  
**DINF/UFPR - Orientadora**

  
Prof.ª Dra. Silvia Modesto Nassar  
**UFSC/DINF - Membro Externo**

  
Prof.ª Dra. Maria Terezinha Steiner  
**UFPR/DMAT - Membro Externo**

  
Prof. Dr. Alexandre Ibrahim Direne  
**DINF/UFPR - Membro Interno**

*A **Deus**, todo poderoso, criador do céu e da terra...Em meio a sua infinita onipotência amou-me com eternidade... A Ele toda Honra, toda Glória e o louvor!*

*A minha família **Itamiro, Dalva, Frã** a quem atribuo a minha essência. Obrigada pela força, e pelo refúgio em todo o tempo. **Maia** , te adoro...*

*Ao meu marido **Marcos Piovesan**, que é minha fonte de amor e inspiração. Obrigada pela paz, incentivo e compreensão que proporciona em todos os momentos da minha vida.*

*Palavras são nada.... Amo vocês!*

## AGRADECIMENTOS

A prof<sup>a</sup>. Dr<sup>a</sup> *Aurora Ramirez Pozo* pela orientação e apoio ao longo do curso realizado.

Ao **curso de pós-graduação em informática** pela oportunidade de realização dentre mestrado.

Aos demais colegas de mestrado pelo convívio cordial e enriquecedor durante o curso.

## SUMÁRIO

<b>1</b>	<b><i>Introdução</i></b> .....	<b>12</b>
1.1	Objetivos do trabalho.....	14
1.2	Organização do trabalho.....	14
<b>2</b>	<b><i>Timetabling</i></b> .....	<b>16</b>
2.1	Conceitos e características gerais.....	16
2.2	Conceito de restrições no contexto de <i>Timetable</i> .....	17
2.3	Variações de <i>Timetabling</i> .....	17
2.3.1	Escala de provas em universidades .....	18
2.3.2	Grades horárias em Instituições de ensino.....	19
<b>3</b>	<b><i>Computação evolutiva</i></b> .....	<b>21</b>
3.1	Algoritmos Genéticos .....	23
3.1.1	Breve histórico.....	23
3.1.2	Características gerais do método.....	24
3.1.3	Terminologia Biológica.....	25
3.1.4	Representação das soluções no espaço de busca.....	27
3.1.5	Inicialização da População .....	27
3.1.6	Operadores Genéticos .....	27
3.1.7	Teorema Fundamental do Algoritmo Genético.....	35
3.1.8	Alguns problemas encontrados em Algoritmos Genéticos .....	36
3.2	Coevolução Cooperativa .....	37
3.2.1	Avaliação do <i>Fitness</i> e escolha de representantes.....	41
3.2.2	Eliminação de populações improdutivas .....	41
3.3	Esquemas Híbridos .....	42
3.4	Algoritmos Meméticos.....	43
<b>4</b>	<b><i>Trabalhos Relacionados</i></b> .....	<b>46</b>
4.1	Heurísticas Dirigidas .....	46
4.2	Grafos Coloridos .....	47

4.3	Programação Lógica .....	47
4.4	Algoritmos Genéticos com penalidades .....	48
4.5	Algoritmos Meméticos.....	49
4.5.1	Representação da Solução para <i>timetabling</i> de exames.....	49
4.5.2	Geração da população inicial.....	49
4.5.3	Avaliação dos indivíduos .....	50
4.5.4	Operadores Meméticos e busca local .....	51
4.5.5	Busca local .....	52
5	<b>Estudo de caso.....</b>	<b>54</b>
5.1	<b>Abordagem Cooperativa para o problema <i>Timetabling</i> .....</b>	<b>54</b>
5.1.1	Modelagem do problema.....	54
5.1.2	Restrições .....	55
5.1.3	Representação dos indivíduos.....	56
5.1.4	Inicialização de populações .....	56
5.1.5	Avaliação de <i>fitness</i> .....	57
5.1.6	Escolha dos representantes .....	58
5.1.7	Elitismo .....	59
5.1.8	Operadores Genéticos .....	59
5.1.9	Colaboração e estagnação entre populações .....	61
5.2	<b>Abordagem Genética para o problema <i>Timetabling</i>.....</b>	<b>61</b>
6	<b>Experimentação e resultados .....</b>	<b>63</b>
6.1	<b>Introdução .....</b>	<b>63</b>
6.1.1	Metodologia.....	63
6.2	<b>Ajuste de parâmetros. ....</b>	<b>66</b>
6.2.1	Tamanho da População .....	66
6.2.2	Taxa de CROSSOVER .....	67
6.2.3	Taxa de mutação .....	69
6.2.4	Taxa Elitismo.....	69
6.2.5	Evolução do <i>Fitness</i> .....	71
6.2.6	Algoritmo Cooperativo X Amostras de disponibilidade de horarios.....	72
6.3	<b>Comparação com Algoritmos Genéticos.....</b>	<b>75</b>
6.3.1	Algoritmo Genético X Amostras de disponibilidade de horarios.....	75

6.4	Contribuições obtidas através do método Coevolutivo .....	77
7	<i>Considerações finais e perspectivas futuras</i> .....	79

## INDICE DE FIGURAS

FIGURA 2.1 - Representação de uma solução de Grades Horárias (CONCILIO, 2000)...	19
FIGURA 3.1- Algoritmo Evolutivo (CONCILIO, 2000) .....	22
FIGURA 3.2 - Representação Gráfica da Roleta de seleção.....	29
FIGURA 3.3 - AG que emprega o conceito de elitismo, extraído de YEPES (2000). ....	30
FIGURA 3.4 - <i>Crossover</i> de um Ponto (MAIA, 2001).....	31
FIGURA 3.5 - <i>Crossover</i> com Dois Pontos (MAIA, 2001) .....	32
FIGURA 3.6 - <i>Crossover</i> Uniforme (CONCILIO, 2000).....	32
FIGURA 3.7 - <i>Crossover</i> PMX (GOLDBERG ,1989).....	33
FIGURA 3.8 -Esquema Gráfico da Ocorrência de Mutação.....	34
FIGURA 3.9 - Convergência Prematura dentro do espaço de busca, extraído de NARDIN (1999). ....	36
FIGURA 3.10 - Modelo Coevolutivo Cooperativo para três espécies (POTTER E DE JONG, 2000).....	39
FIGURA 3.11 - Algoritmo Cooperativo, extraído e adaptado de COSTA e BRUNNA (2000). ....	40
FIGURA 3.12 - Ciclo evolutivo Memético (NEWALL, 2000).....	44
FIGURA 3.13 – Adição de busca local a ciclos evolutivos (NEWALL, 2000).....	45
FIGURA 5.1- Representação do cromossomo. ....	56
FIGURA 5.2 - <i>Crossover</i> implementado. ....	59
FIGURA 5.3 - Representação de um Cromossomo em AG's .....	62
FIGURA 6.1 - Número de Gerações em função da variação da taxa de <i>crossover</i> .....	68
FIGURA 6.2 - Número de gerações em função da variação da taxa de mutação. ....	69
FIGURA 6.3 - Número de Gerações em função da variação da taxa de elitismo. ....	70
FIGURA 6.4 – Evolução de <i>fitness</i> para a amostra 6. ....	71
FIGURA 6.5 - Número de Gerações obtido na Evolução Cooperativa frente a diferentes níveis de complexidade na busca .....	73
FIGURA 6.6 - Representação Cooperativa de um indivíduo para <i>timetable</i> para exames. ....	78



## INDICE DE TABELAS

TABELA 3.1-Termos utilizados em AG's e seus significados na biologia. ....	26
TABELA 3.2 - Dados ilustrativos para Seleção por Roleta.....	28
TABELA 4.1 - Codificação da solução para <i>timetable</i> (BURKE e NEWALL, 1996).....	49
TABELA 6.1- Número de gerações obtido com variação do tamanho da população .....	67
TABELA 6.2 - Populações estagnadas X Amostras de disponibilidade horária.....	75
TABELA 6.3- <i>Fitness</i> AG X Amostras de disponibilidade horária. ....	76

## RESUMO

A produção de grades horárias em instituições de ensino é uma tarefa complexa e de difícil solução, pois, neste contexto, existem muitas restrições necessárias à validade e aplicabilidade das respostas produzidas. Na literatura, a produção de grades horárias é, na verdade, uma das variações de *timetabling*, o qual, em essência, é um problema de escalonamento de eventos em um período finito de tempo, sujeito a restrições, como por exemplo, tempo, recursos humanos disponíveis (professores), recursos físicos existentes (salas de aula) e atividades a serem desenvolvidas (exames, aulas, entre outros). Para solucionar esse problema e automatizar o processo, abordagens de Inteligência Artificial têm sido aplicadas com sucesso, mais especificamente, os métodos da Computação Evolutiva. A computação evolutiva define uma classe de algoritmos que modelam computacionalmente os conceitos da teoria da Evolução de Charles Darwin. Esses algoritmos aplicam operadores genéticos sobre populações de indivíduos, visando à produção de indivíduos mais aptos que os antigos. Como resultado, obtêm-se indivíduos ou soluções candidatas com um alto grau de aptidão para solucionar um problema específico. O objetivo principal deste trabalho é estudar e implementar uma solução para o problema de Geração de Grades Horárias, com base na Computação Evolutiva. O método evolutivo escolhido é denominado Algoritmo Coevolutivo Cooperativo. Esse método subdivide um problema complexo em problemas menores, sendo que cada um deles é representado por uma população pertencente ao domínio do problema. Cada uma dessas populações possui características individuais e, no processo, todas evoluem paralelamente, de maneira cooperativa, por meio de sucessivas aplicações de operadores genéticos. Ao final do processo, os representantes de cada uma das populações formam, em conjunto, uma solução completa. Para verificar a validade do método para a resolução do problema em estudo, implementou-se um algoritmo cooperativo. Os resultados dos experimentos mostraram que algoritmos cooperativos são ferramentas poderosas, capazes de resolver problemas complexos de otimização numérica sujeitos a restrições.

**Palavras-chave:** Algoritmos Evolutivos, Algoritmos Coevolutivos Cooperativos, *Timetabling*, Otimização Numérica, Produção de Grades Horárias.

## ABSTRACT

The production of schedules in educational institutions is a complex and hard solution, because in this scenario there are many constraints required to the validity and applicability of the produced results. According to the literature, the production of schedules in educational institutions, in fact, one of the *Timetabling* variations, which is, in essence, a problem of event scheduling in a finite period of time, subjected to constraints, for example, time, human resources (teachers, lecturers, etc), physical resources (rooms) and activity to be advanced (exams, class, etc). In order to solve this problem and automate the process, Artificial intelligence methods have been used with success, more specifically, Evolutionary Computing methods. The evolutionary computing defines a class of algorithms that computationally model the Charles Darwin's Evolution theory concepts. These algorithms apply genetic operators over a population of individuals, aiming at the production of individuals more capable than the older ones. The result is candidate individuals or solutions with a high degree of capability to solve a specific problem. The main goal of this work is to study and implement a solution for the Schedule Generation problem, based on Evolutionary Computing. The chosen evolutionary method is denominated Cooperative Co-Evolutionary Algorithm. This method divides a complex problem in smaller ones, being each small problem represented by a population that belongs to the problem domain. Each population has individual features and all of them evolve in a parallel and cooperative way through the successive application of genetic operators. At the end of this process, the representatives of each population compose a complete solution. In order to verify the validity of the method for the solution of the problem under study, a cooperative algorithm was implemented. The results of the experiments showed that the co-evolutionary algorithms are powerful tools, capable of solving complex problems of numeric optimization subjected to constraints.

**Keywords:** Evolutionary Algorithms, Co-Evolutionary Cooperative Algorithms, *Timetabling*, Numeric Optimization, Schedule Generation.

# 1 INTRODUÇÃO

Segundo GOLDBERG (1989), Charles Darwin estudando as espécies e sua evolução, coletou durante anos uma grande quantidade de material que demonstrou a existência de inúmeras variações genéticas em cada espécie. Estes estudos, associados às pesquisas de outros cientistas, tornaram evidente que espécies animais efetivamente se modificam. As novas espécies são produzidas através da seleção natural que ocorre pela pressão ambiental ou competição por recursos do meio necessários à sobrevivência. Desta seleção restam os indivíduos que forem mais preparados ou adaptados sendo estes os que têm maiores chances de sobreviver e posteriormente se reproduzir.

A Computação Evolutiva foi desenvolvida com base na teoria evolucionista de Darwin e compreende uma série de métodos que modelam computacionalmente os mecanismos básicos da teoria da evolução natural. Em essência os métodos evolutivos trabalham com uma população de indivíduos. A população de indivíduos é processada em um ciclo evolutivo que tem como objetivo atingir uma solução ótima entre as soluções candidatas através da melhora gradativa dos cromossomos dos indivíduos. Em função dos recursos proporcionados pela abordagem evolutiva, sua principal aplicação se dá a problemas de otimização numérica.

Os problemas classificados como de otimização numérica consistem de uma função objetivo que precisa ser otimizada, e restrições impostas ao espaço de busca. A solução do problema é portanto, um ou vários pontos ótimos no espaço. O problema de otimização a ser analisado neste trabalho é conhecido como *Timetabling*. Este problema apresenta múltiplas restrições e grande número de variáveis que devem ser atendidas na obtenção da solução ótima.

Algumas técnicas convencionais já foram utilizadas anteriormente na resolução de *timetabling* tais como: Heurísticas Dirigidas, Grafos Coloridos, Programação Lógica entre outros. Dentre os métodos derivados da abordagem evolutiva estão os Algoritmos Genéticos (AG's). Contudo, AG's são tradicionalmente utilizados em problemas de

otimização que não apresentem restrições aplicadas às soluções candidatas. Uma evidência disto é que a aplicação de operadores genéticos convencionais, utilizados durante o processo evolutivo, não tratam diretamente restrições.

POTTER e DE JONG (2000) propuseram o método de Algoritmos Coevolutivos Cooperativos, que são uma extensão dos AG's, explorando o potencial desta abordagem na resolução de problemas de otimização e viabilizam a factibilidade para soluções encontradas mesmo em problemas complexos. Neste método, um problema (ecossistema) é dividido em problemas menores (espécies). Assim, a arquitetura definida pelo método coevolutivo modela um ecossistema consistindo de duas ou mais espécies. As espécies interagem entre si dentro de um modelo de domínio compartilhado e têm um relacionamento cooperativo a fim de encontrar uma solução para o problema complexo.

A principal motivação para a utilização da abordagem evolutiva baseia-se na argumentação de CONCILIO (2000), explicando que para problemas complexos de otimização tem-se uma explosão combinatória de candidatos à solução dentro de um espaço de busca. Desta forma buscas exaustivas promovidas por métodos convencionais, não conseguiriam atingir uma solução ótima, caracterizando portanto o problema como computacionalmente intratável.

Além disso, a abordagem Coevolutiva promove a factibilidade dos indivíduos produzidos pelo processo evolutivo através da divisão de espécies. Assim, indivíduos altamente especializados ou com um valor baixo de *fitness* não podem prejudicar indivíduos de outra espécie, pois as mesmas são evoluídas de forma isolada. Algoritmos Coevolutivos, apresentam portanto, grande potencial para a resolução de problemas complexos de otimização.

O objetivo deste trabalho consiste em estudar e verificar o comportamento do método de Evolução Cooperativa para o caso específico de geração de grades horárias para uma universidade. Para que este objetivo seja viabilizado, o algoritmo Cooperativo será implementado. Além disso um algoritmo genético clássico também será desenvolvido para que comparativos entre ambas as técnicas possam ser realizados.

## 1.1 Objetivos do trabalho

O presente trabalho tem como objetivo geral investigar a aplicação de Algoritmos Coevolutivos Cooperativos na otimização de problemas de escalonamento (*scheduling*) sujeitos à restrições, sendo enfocado neste trabalho *timetabling*.

Como objetivos específicos, pode-se destacar:

- Observação do problema, bem como sua modelagem de acordo com a formalização evolutiva;
- Investigação da abordagem Evolutiva, Algoritmos Genéticos e Algoritmos Cooperativos bem como métodos não evolutivos anteriormente utilizados na resolução de *timetabling*;
- Implementação de um sistema, para a produção automática de grades horárias no curso de bacharelado em informática da UFPR utilizando a abordagem de Algoritmos Cooperativos;
- Implementação de um sistema para o problema da produção de grades horárias utilizando Algoritmos Genéticos Clássicos;
- Comparação dos resultados obtidos por Algoritmos Genéticos Clássicos e Algoritmos Cooperativos para o problema em estudo.

## 1.2 Organização do trabalho

Este texto está organizado em seis capítulos. O presente capítulo trata de considerações gerais sobre o trabalho, em nível introdutório. Já o capítulo dois apresenta o problema objeto de estudos do trabalho conhecido como *timetabling*. O problema é apresentado sob forma de conceito, elementos, formalização e variações mais comuns.

No terceiro capítulo tem-se a apresentação da teoria e fundamentos da Computação Evolutiva enfocando a técnica de Algoritmos Genéticos, Algoritmos Híbridos e Algoritmos Cooperativos.

O quarto capítulo apresenta alguns trabalhos relacionados como o problema bem como a solução proposta por NEWALL (2000) através de Algoritmos Meméticos. Neste

capítulo também encontra-se descrito o algoritmo Cooperativo implementado para a solução de *timetabling*, no contexto de Geração automática de Grades Horárias .

O quinto capítulo foi reservado para a apresentação de um estudo de caso para a resolução do qual foram desenvolvidos um algoritmo Coevolutivo e um algoritmo Genético. Nesta mesma seção os resultados obtidos pela aplicação do método coevolutivo serão apresentados, bem como um comparativo desta abordagem com Genéticos. O último capítulo foi destinado a considerações finais e conclusões obtidas neste trabalho.

## 2 TIMETABLING

### 2.1 Conceitos e características gerais

De acordo com ROSS *et al.* (1999,p.2), “*Timetabling* é em essência um problema de escalonamento sujeito a restrições onde os eventos são considerados como instâncias ou recursos do problema, ou ainda, coleção de eventos que devem acontecer dentro de um período finito de tempo.” Assim, recursos não podem ser solicitados por dois ou mais eventos ao mesmo tempo ou devem existir em uma quantidade suficiente para servir todos os eventos durante todo o tempo de escalonamento.

Segundo (ROSS *et al.*, 2000, p.1), um *timetable* genérico pode ser definido por três conjuntos básicos. São eles:

- $E = \{e_1, e_2, \dots, e_v\}$ , ou seja, um conjunto finito de eventos que inclui atividades diversas como exames, seminários, projetos, ou aulas;
- $T = \{t_1, t_2, \dots, t_s\}$ , que é um conjunto finito de horários, para realização dos eventos e
- $A = \{a_1, a_2, \dots, a_m\}$ , que implica em um conjunto finito de agentes (instrutores, monitores ou professores), que têm o papel de monitorar eventos particulares.

Baseando-se nesta definição anterior é válido representar o conjunto de elementos envolvidos no contexto de *timetabling* pelo conjunto  $\{e, t, a\}$  onde  $e \in E$  (conjunto de eventos ou atividades),  $t \in T$  (conjunto de horários) e  $a \in A$  (conjunto de recursos ou agentes), podendo ser interpretado como : o evento  $e$  inicia em um tempo  $t$  e tem como agente  $a$ .

Conclui-se, portanto que uma *timetable* nada mais é do que uma coleção de triplas como o descrito, uma por evento, sendo que as escalas a serem produzidas não devem violar um conjunto de restrições pré-definido pelo contexto.



## 2.2 Conceito de restrições no contexto de *Timetable*

As restrições a que um problema de escalonamento clássico, neste caso *timetabling*, está sujeito podem ser entendidas como situações que estabelecem um padrão de qualidade ou definem regras para a produção destas tabelas. Como exemplo, pode-se citar duas restrições básicas presentes em um *timetable* genérico que garantem a factibilidade das tabelas produzidas:

- Nenhuma entidade pode ser solicitada por mais de um local ao mesmo tempo;
- O recurso solicitado pelos eventos escalonados no período não pode exceder os recursos disponíveis para cada um dos períodos na *timetable*.

Portanto, as restrições estão presentes no contexto e a partir do qual será produzido um *timetable*, variam de acordo com os objetos inclusos. E quando as *timetables*, produzidas violam restrições, as tabelas produzidas podem apresentar-se infactíveis ou não aplicáveis.

Segundo NEWALL (2000), os tipos principais de restrições aplicáveis a este problema podem ser classificadas como *hard*, quando devem ser essencialmente satisfeitas para produção de uma *timetable* factível, ou *soft*, que são restrições desejáveis, mas não essenciais.

Esta diferença permite observar o conceito de *qualidade* de *timetable*, onde condições essenciais para uma tabela factível são satisfeitas e violações de restrições do tipo *soft* são minimizadas até um nível aceitável. O termo aceitável sempre vai estar dependente do montante de recursos disponíveis.

## 2.3 Variações de *Timetabling*

*Timetabling* pode ser aplicado a diversos contextos, e para isso basta modificar as variáveis e as restrições envolvidas no problema. Algumas variações de *Timetabling* podem ser descritas como: Escalonamento de funcionários em turnos, Remanejamento de máquinas em fábrica, tabelas de horários ou recursos (exames, salas de aula, entre outros) para escolas ou para universidades. Assim, o problema é caracterizado por uma natureza fixa e um conjunto de restrições variáveis.

Nas próximas seções serão vistas duas variações comuns para *timetabling*. Na primeira variação tem-se o escalonamento para provas em universidades e a segunda consiste da produção de grades horárias em instituições de ensino, que é o objeto de estudo deste trabalho.

### 2.3.1 ESCALA DE PROVAS EM UNIVERSIDADES

Para escalonamento de exames ou provas em instituição de ensino superior, NEWALL (2000) explica que recursos como exames, salas de aula, professores, entre outros devem ser agrupados, ou escalonados, em função das necessidades de cada curso, ano e programa.

Neste contexto tem-se duas restrições básicas: nenhum estudante pode fazer mais de uma prova ao mesmo tempo e nenhuma sala de aula pode comportar mais alunos que carteiras existentes quando da aplicação de exames.

BURKE e NEWALL (1996) argumentam que as restrições acima definem regras de produção de *timetables* válidas. Porém é usualmente interessante produzir tabelas com boa qualidade, ou seja, que sejam na realidade práticas e deixem os usuários satisfeitos. Para tanto se faz necessário o acréscimo das seguintes restrições:

- Nenhum estudante deve ter dois exames em períodos adjacentes (sequenciais);
- Nenhum estudante deve ter dois exames no mesmo dia;
- O exame A deve ser escalonado antes do exame B e podem ser escalonados ao mesmo tempo somente quando os dois exames possuírem o conteúdo igual ou similar;
- Caso um exame A necessite de recursos especiais para ser aplicado, e estes recursos estiverem disponíveis somente em determinada sala, este exame deve ser conduzido nesta sala.

### 2.3.2 GRADES HORÁRIAS EM INSTITUIÇÕES DE ENSINO

Em termos gerais, a produção de tabelas de horário, consiste da alocação de recursos para produção de uma grade horária semanal que devem ser instanciados com professores e disciplinas.

Este problema é conhecido na Inteligência Artificial como Satisfação de Restrições. Neste tipo de problema, o processo de busca da solução tem o objetivo de encontrar um estado, dentro do espaço de busca que satisfaça um conjunto de restrições.

Em CONCILIO (2000), é possível se observar um exemplo de grade horária, representado pela figura 2.1. Esta figura é composta por uma única turma que cumpre cinco dias (segunda a sexta-feira) e cinco horários semanais de aula. As restrições existentes neste exemplo são as seguintes:

- Escalonar cada um dos 25 professores uma única vez durante toda a semana;
- O número de aulas em um mesmo dia não deve exceder cinco;
- Quando duas aulas da mesma disciplina acontecerem em um mesmo dia, estas devem ocorrer de forma sequencial ou germinada.

FIGURA 2.1 - Representação de uma solução de Grades Horárias (CONCILIO, 2000)

Segunda-feira	<i>W</i>	<i>P</i>	<i>F</i>	<i>R</i>	<i>A</i>
Terça-feira	<i>E</i>	<i>S</i>	<i>N</i>	<i>K</i>	<i>L</i>
Quarta-feira	<i>D</i>	<i>I</i>	<i>G</i>	<i>X</i>	<i>U</i>
Quinta-feira	<i>B</i>	<i>V</i>	<i>T</i>	<i>J</i>	<i>H</i>
Sexta-feira	<i>Y</i>	<i>O</i>	<i>C</i>	<i>M</i>	<i>Q</i>

Conforme mostra a figura 2.1, cada indivíduo da população é representado por uma matriz onde as linhas são representadas por dias da semana, e as colunas apresentam os diferentes horários do dia que deverão ser preenchidos com recursos disponíveis no problema, neste caso, as disciplinas representadas por letras de A a Y.

Existem na literatura diversos métodos que foram utilizados para resolver variantes do problema *timetabling*. Entre estes métodos encontram-se aqueles descritos pela

Computação Evolutiva. No próximo capítulo, apresenta-se a descrição conceitual desta abordagem bem como alguns métodos desenvolvidos a partir da mesma.

### 3 COMPUTAÇÃO EVOLUTIVA

NARDIN (1999) afirma que na natureza cada organismo possui cromossomos, genes, exons, íntrons e códons, constituindo um sistema genético. Um determinado grupo de indivíduos vive em conjunto, constituindo uma população. Nesta população há os organismos melhores, que são os que têm mais chances de acasalar e gerar bons descendentes. Estes descendentes são mais bem adaptados do que a média da população, pois receberam melhores genes. Ao final, vence a lógica, de sobreviver o mais adaptado ao nicho ecológico da população. Este sistema de escolha é passado às gerações subsequentes, melhorando cada vez mais as populações envolvidas.

Segundo CONCILIO (2000, p.3) “o termo Computação Evolutiva surgiu em 1991. Representa um esforço para unir os pesquisadores que trabalhavam com simulação computacional nas áreas ligadas à evolução”. ZIZTLER (1999, p.5) argumenta que “O termo Algoritmo Evolutivo padroniza uma classe de métodos estocásticos de otimização que simulam o processo da evolução natural.”

“A computação evolutiva, juntamente com os sistemas *fuzzy*, redes neurais artificiais e agentes autônomos, está posicionada como um sub-item da Inteligência Artificial. Sabe-se também que não garante eficiência total na obtenção da solução, mas geralmente alcança uma boa aproximação para a solução ótima”. (CONCILIO 2000, p. 09)

Desde 1970 várias metodologias evolucionárias foram propostas, tais como: Algoritmos genéticos, Estratégias evolutivas e Programação Evolutiva. Estas abordagens operam com um conjunto de soluções candidatas e modificam esses conjuntos através da aplicação de dois princípios básicos evolutivos: a *seleção*, representando a competição por recursos entre seres vivos onde os melhores são mais aptos à sobrevivência e *variação* que representa a capacidade de reprodução de novos seres pela recombinação e mutação.

Computacionalmente, estes processos naturais são representados em forma de operadores genéticos (recombinação e mutação), instrumentos avaliativos (que atribuem um valor de aptidão ou *fitness* aos indivíduos) e métodos seletivos para os indivíduos da

população. Estes processos são utilizados para promover o ciclo evolutivo apresentado na figura 3.1.

ZITTLER (1999, p.4) afirma que: “Embora os princípios empregados nos Algoritmos Evolucionários serem simples, estes algoritmos têm provido um geral, robusto e poderoso mecanismo de busca.”

CONCILIO (2000) resumidamente, destaca como características das principais abordagens evolutivas:

1) Algoritmos Genéticos: Desenvolvem um modelo computacional formalizando os processos adaptativos naturais para resolução de problemas de otimização sendo que para busca de soluções utiliza-se de operadores tais como mutação, seleção entre outros;

2) Programação Genética: É aplicável à uma grande variedade de problemas onde o código genético é representado como uma árvore de atributos e tem por objetivo evoluir programas de computador usando princípios da evolução natural;

3) Estratégias Evolutivas: Propostas inicialmente com o objetivo de solucionar problemas de otimização de parâmetros, tanto discretos como contínuos, empregando apenas o operador de mutação .

4) Programação Evolutiva: Emprega apenas o operador de mutação e é aplicável à problemas de otimização, com pequenas diferenças, nos procedimentos de seleção e codificação de indivíduos, em relação a técnica de Estratégias Evolutivas.

FIGURA 3.1-Algoritmo Evolutivo (CONCILIO, 2000)

```
Procedimento programa evolutivo  
início  
t <- 0  
inicializar  $P(t)$   
avaliar  $P(t)$   
enquanto não (condição de parada) faça  
    t = t+1  
    selecionar  $P(t)$  a partir de  $P(t-1)$   
    alterar  $P(t)$   
    avaliar  $P(t)$   
fim.
```

O algoritmo evolucionário descrito na figura 3.1 é baseado nos conceitos acima descritos e a evolução natural aqui é representada pelo processo computacional iterativo. Primeiro, uma população inicial é criada randomicamente (ou de acordo com algum esquema predefinido) sendo o ponto inicial do processo de evolução. Este processo é seguido pela avaliação dos indivíduos que atribui o valor de *fitness*. Então um laço consistindo de passos de seleção, recombinação e/ou mutação e avaliação (atribuição de *fitness*), é executado um certo número de vezes. Cada iteração do laço é chamada geração, e o número máximo de gerações pode ser pré-definido, servindo como critério de parada.

Outras condições como estabilização da população ou existência de um indivíduo com suficiente qualidade podem também ser usadas como critérios de parada. Ao final, o melhor indivíduo da população é definido como solução para o problema que está sendo resolvido.

Neste capítulo são abordados vários métodos derivados da Computação Evolutiva como algoritmos Genéticos, algoritmos Cooperativos, algoritmos Híbridos e algoritmos Meméticos. O enfoque desta seção será dado aos aspectos conceituais de cada abordagem. Como algoritmos evolutivos são amplamente aplicáveis a problemas de otimização, o objetivo desta seção é apresentar os recursos e vantagens oferecidos pela abordagem na resolução destes problemas.

### **3.1 Algoritmos Genéticos**

#### **3.1.1 BREVE HISTÓRICO**

Os AG's foram inicialmente desenvolvidos pelo professor John Holland, da Universidade de Michigan, nos EUA, em suas explorações dos processos adaptativos de sistemas naturais e suas possíveis aplicabilidades em projetos de *softwares* de sistemas artificiais.

O método foi formalmente introduzido no seu livro *Adaptation in Natural and Artificial System* (HOLLAND, 1975). Convém ainda salientar que a idéia dos Algoritmos Genéticos é mais antiga, como reconhece o próprio Holland, referindo-se a trabalhos anteriores e ainda outras abordagens semelhantes.

Segundo GOLDBERG (1989) o trabalho de Holland teve duas metas principais:

- abstrair e rigorosamente explicar os sistemas adaptativos naturais e
- projetar *softwares* de sistemas artificiais que assegurassem mecanismos importantes de seleção natural.

Nesta obra, apresentam-se quatro principais diferenças entre algoritmos evolutivos e técnicas convencionais:

1. Trabalhar com a codificação de um conjunto de parâmetros, não com os próprios parâmetros;
2. Buscar em uma população de pontos, não em um ponto único;
3. Utilizar informações de custo e não derivadas ou outro conhecimento auxiliar;
4. Usar regras de transição aleatórias e não regras determinísticas.

### 3.1.2 CARACTERÍSTICAS GERAIS DO MÉTODO

Os AG's diferem da maioria dos algoritmos convencionais de otimização, pois os últimos utilizam um único ponto no espaço de busca, sendo que uma regra de transição determina o próximo ponto. O que ocorre nestes métodos, chamados ponto-a-ponto, é que existe a possibilidade do processo de busca ficar “preso” a máximos locais, sem considerar todo o espaço de busca. Os AG's exploram o espaço de busca utilizando diversos pontos, tendo a possibilidade de encontrar muitos pontos bons em paralelo diminuindo assim a probabilidade da busca ficar presa a ótimos locais.

Segundo CONCILIO (2000) o processo de evolução executado por um AG corresponde a um procedimento de busca em um espaço de soluções potenciais para alcançar um objetivo proposto, e essa busca requer um equilíbrio entre o aproveitamento das melhores hipóteses candidatas e exploração do espaço de busca.

Assim, os componentes essenciais de um algoritmo genético são:

- a representação genética para soluções candidatas ou potenciais (codificação);
- uma maneira de criar uma população inicial de soluções candidatas ou potenciais;



- uma função de avaliação que faz o papel da pressão do ambiente, classificando as soluções em termos de sua aptidão;
- o emprego de operadores genéticos e valores para os parâmetros usados pelo algoritmo, tais como tamanho da população, taxas de probabilidade da aplicação de operadores genéticos, entre outros.

Segundo NARDIN (1999), em geral os AG's têm sido usados para solução de funções de otimização, mostrando-se bastante eficientes e confiáveis. Mas, nem todos os problemas são resolvidos de forma satisfatória por não poderem ser representados adequadamente pelos mecanismos da abordagem.

Antes de aplicar AG para a resolução de determinados problemas, é necessário verificar a possibilidade de representar as seguintes características do mesmo:

- O espaço de busca de possíveis soluções para o problema, delimitado-o dentro de uma certa faixa de valores;
- Uma função objetivo, que indique o quanto uma determinada resposta é boa ou ruim;
- As soluções, permitindo sua codificação de forma computacional.

### 3.1.3 TERMINOLOGIA BIOLÓGICA

Segundo GOLDBERG (1989), a teoria da evolução prediz que o meio ambiente seleciona, de cada geração, os seres vivos mais aptos para sobrevivência. Durante a existência dos indivíduos, ocorrem fenômenos como mutação e *crossover*, que atuam sobre o material genético armazenado nos cromossomos. Estes processos promovem a variabilidade genética dos seres vivos na população.

Em AG's, os processos expressam-se como metáfora à estes fenômenos, explicando assim existência de muitos termos originados da biologia, tais como apresentados na tabela abaixo:

TABELA 3.1-Termos utilizados em AG's e seus significados na biologia.

TERMO	BIOLOGIA	AG'S
<b>Gen ou Gene</b>	Unidade de hereditariedade que é transmitida pelo cromossomo e que controla as características do organismo.	Parâmetro codificado no cromossomo, ou seja, um elemento do vetor que representa o cromossomo.
<b>Genoma ou Cromossomo</b>	Conjunto completo de genes de um organismo. Genoma é o conjunto de vários cromossomos.	Estrutura de dados que codifica uma solução para um problema, ou seja, um cromossomo representa um único ponto no espaço de busca.
<b><i>Fitness</i></b>	Valor que indica o grau de aptidão do indivíduo ao meio que está inserido	Fornece para cada indivíduo a medida de quão próximo a uma solução considerada satisfatória se encontra.
<b>Indivíduo</b>	Um membro da população.	Um indivíduo é formado pelo cromossomo (conjunto de genes) e por um valor de <i>fitness</i> .
<b>Genótipo</b>	Composição genética contida no genoma.	Informação contida no cromossomo.
<b>Fenótipo</b>	Características visíveis, como cor dos olhos, dos cabelos e outras características físicas	Objeto, estrutura ou organismo constituído a partir das informações do genótipo representando a codificação do cromossomo
<b>Alelo</b>	Formas alternativas do gene	Valores que o gene pode assumir. Por exemplo, um gene que representa o parâmetro cor de um objeto poderia ter o alelo azul, preto, verde, etc.

O ciclo desenvolvido pelos algoritmos genéticos é idêntico ao apresentado na figura 3.1. Nesta seção, serão descritos os passos compreendidos por este ciclo e também alguns aspectos que influenciam diretamente no desempenho do mesmo.

#### 3.1.4 REPRESENTAÇÃO DAS SOLUÇÕES NO ESPAÇO DE BUSCA

Tradicionalmente em AG's, as soluções candidatas também denominadas indivíduos, são representados genotipicamente por vetores binários. Cada elemento destes vetores denota a presença (1) ou ausência (0) de uma determinada característica equivalendo ao seu genótipo. Estes elementos podem ser combinados formando as características reais do indivíduo, ou o seu fenótipo.

Portanto cada indivíduo conterà o fenótipo, o genótipo, o *fitness*, conhecido também como valor de aptidão, além de outras informações auxiliares de cada indivíduo em particular. Ao conjunto de indivíduos a serem submetidos ao ciclo evolutivo dá-se o nome de população.

#### 3.1.5 INICIALIZAÇÃO DA POPULAÇÃO

Segundo CONCILIO (2000), o método mais comumente empregado na inicialização da população é a geração aleatória dos indivíduos. Porém o conhecimento a partir do problema poderá ser utilizado para melhorar o desempenho da criação dos indivíduos já na geração inicial.

Em problemas com restrições deve-se tomar o cuidado de não gerar indivíduos inválidos na etapa de inicialização. Este fato poderia retardar o processo evolutivo de uma população que é tratada em um espaço de busca sujeito a muitas restrições.

#### 3.1.6 OPERADORES GENÉTICOS

Os operadores genéticos estendem a busca até atingir resultado satisfatório. Estes operadores transformam a população através das gerações e são necessários para que a população possa se diversificar, mantendo características adaptativas adquiridas pelas gerações anteriores.

É possível classificar os operadores genéticos como de seleção, mutação e *crossover* (ou cruzamento).

## Métodos de Seleção

É necessário, depois de definir a representação dos indivíduos, escolher o método que irá promover a seleção dos mesmos para gerar descendentes. A proposta deste processo é privilegiar os melhores indivíduos na população esperando de sua descendência também um alto *fitness*.

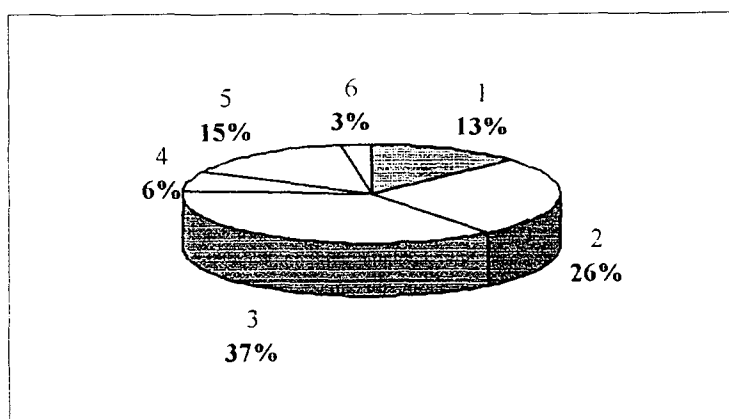
Segundo COSTA e BRUNA (2002), quando a seleção de indivíduos ocorre de forma restrita significa que indivíduos de *fitness* alto dominarão a população, reduzindo a diversidade necessária para futuras mudanças e progressos. Porém, se for realizada de forma abrangente, o processo evolutivo pode ocorrer de forma muito lenta em consequência do *fitness* baixo apresentado pelos indivíduos da população.

Existem vários métodos de seleção, sendo que o mais utilizado é denominado *Roulette wheel*, ou Método da Roleta. Este método foi inicialmente proposto GOLDBERG (1989) e consiste da criação de uma Roleta onde cada cromossomo possui um segmento proporcional ao seu *fitness*. Assim, quanto maior for o *fitness* apresentado por determinado indivíduo, maior será a área por ele ocupada no domínio da roleta promovendo assim maior probabilidade de ser sorteado. A figura 3.2 apresenta o funcionamento do método de Roleta.

TABELA 3.2 - Dados ilustrativos para Seleção por Roleta

Nº do cromossomo	String	<i>Fitness</i>	Porcentagem do total
1	0101101	45	13
2	1011001	89	26
3	1111101	125	37
4	0010101	21	6
5	0110100	52	15
6	0001001	9	3
Total		341	100.0

FIGURA 3.2 - Representação Gráfica da Roleta de seleção



Na tabela 3.2 tem-se uma população com 6 cromossomos cujo *fitness* (terceira coluna) é dado apenas pela conversão do código binário em decimal. Usando os valores apresentados na quarta coluna é elaborada a roleta mostrada na figura 3.2. Esta roleta será girada *n* vezes para efetuar a seleção da população auxiliar que irá gerar alguns descendentes.

Pode-se observar que cada indivíduo de uma determinada geração recebe uma probabilidade de passar para a próxima geração proporcional ao seu *fitness*, medido em relação ao *fitness* total da população.

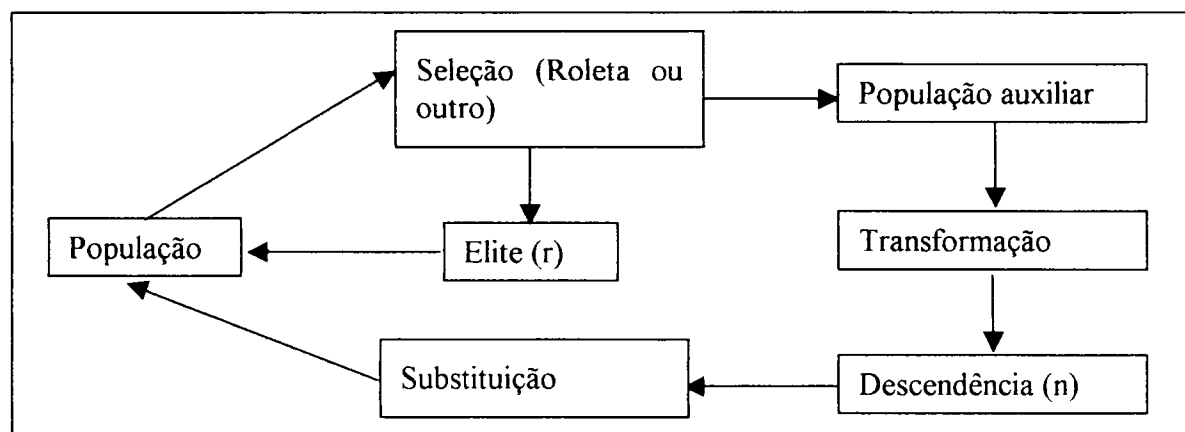
Na figura 3.2 deve-se considerar que a região equivalente ao cromossomo 1 tem início em 0% e vai até 13%, a região equivalente ao cromossomo 2 tem início em 13% e vai até 39% e, assim sucessivamente. Portanto, se durante a seleção da roleta, o ponto sorteado for o que pertence a região do cromossomo 3 este será o escolhido. Assim, é possível perceber que indivíduos que têm maior valor de *fitness* têm maiores chances de serem selecionados, já que a faixa correspondente é maior.

Os melhores indivíduos podem também ser simplesmente perdidos, caso a roleta não pare na faixa correspondente por ser um processo aleatório. Um método consistente seria escolher como solução o melhor indivíduo encontrado ao longo de todas as gerações ou apenas manter sempre o melhor indivíduo da geração atual na geração seguinte. Este processo é conhecido como seleção elitista.

Existe também o método de seleção por Torneio. Neste tipo de seleção,  $n$  indivíduos que definem o tamanho do torneio são escolhidos aleatoriamente na população. Um número aleatório  $r$  é escolhido entre 0 e 1. Se  $r < k$  (onde  $k$  é um parâmetro pré determinado), o melhor dos indivíduos é selecionado para ser um pai; de outra forma, o indivíduo com *fitness* menor é selecionado, então são devolvidos para a população original e podem ser selecionados novamente.

O elitismo, segundo MITCHELL (1998, p. 168), “é uma adição a muitos métodos de seleção, que forçam o AG a manter alguns dos melhores indivíduos a cada geração”. Estes indivíduos continuam a contribuir com seu código genético na produção de descendentes que irão compor as próximas gerações. O método é útil, pois tais indivíduos podem ser perdidos se não forem selecionados para reprodução ou podem ser destruídos por *crossover* ou mutação. O esquema gráfico proposto por YEPES (2001) é mostrado na figura 3.3 para um AG que utiliza elitismo.

FIGURA 3.3 - AG que emprega o conceito de elitismo, extraído de YEPES (2000).



A figura 3.3 demonstra que em AG's que utilizam seleção elitista os indivíduos mais aptos são mantidos em uma próxima geração imediatamente após o processo de seleção. O elitismo melhora muito o desempenho de AG's entretanto perde-se um pouco em termos de diversidade de indivíduos, uma vez que os mantém de forma intacta durante diversas gerações.

## Operador de Cruzamento (*Crossover*)

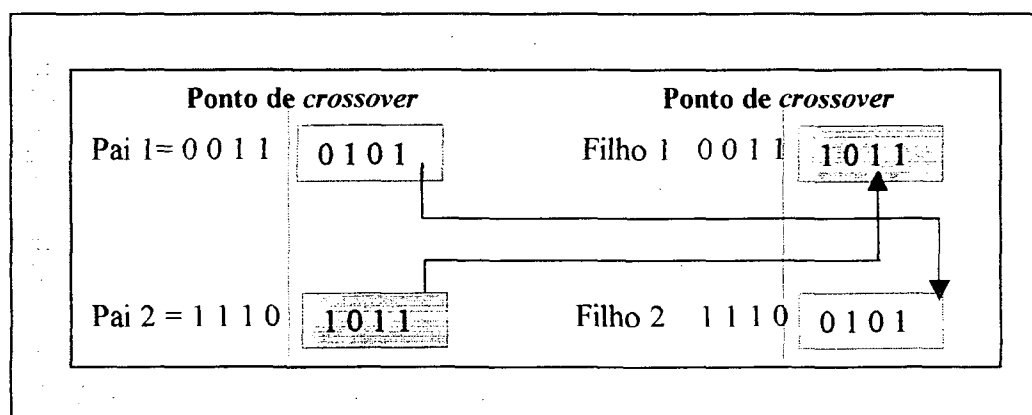
O operador *crossover* é um operador binário e é aplicado aos indivíduos de acordo com uma probabilidade dada pela taxa de cruzamento  $P_c$ . Este operador é utilizado em indivíduos escolhidos pelo operador de seleção e promove a troca de genes ou informações entre os mesmo para geração de novos indivíduos.

Assim, cada par de cromossomos dá origem a novos indivíduos, que formarão a população da próxima geração. Como o *crossover* é considerado o operador genético predominante, a taxa  $P_c$  deve ser maior que a taxa de mutação.

Os métodos mais usuais de *crossover* são: de um ponto, de dois pontos e uniforme. GOLDBERG (1989) ainda sugere um outro tipo de cruzamento conhecido como *cruzamento parcialmente casado* (PMX), aplicável a problemas onde a ordem com que os genes aparecem no cromossomo influencia diretamente no valor de *fitness* dos indivíduos.

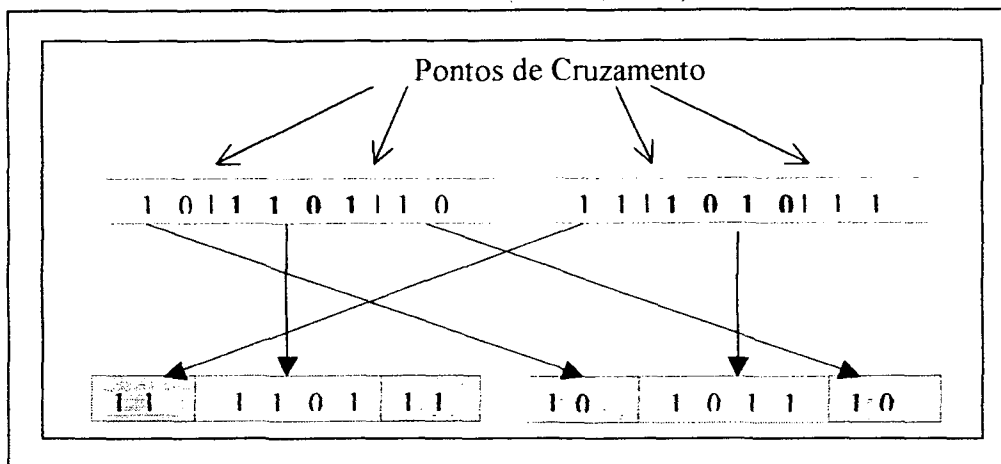
No *crossover de um ponto*, um ponto de corte aleatório é escolhido entre os genes do indivíduo. A partir deste ponto, as informações genéticas dos pais são trocadas de forma que informações anteriores ao ponto escolhido em um dos pais são ligadas às informações posteriores a este ponto no outro pai. Pode-se exemplificar um ponto de cruzamento como segue. Supondo dois pais:  $P1 = 00110101$  e  $P2 = 11101011$  se o ponto 4 for escolhido, então, os filhos terão as configurações como mostra a figura 3.4.

FIGURA 3.4 - *Crossover* de um Ponto (MAIA, 2001).



O processo desenvolvido pelo *crossover de dois pontos* é similar ao anterior, porém são sorteados aleatoriamente dois pontos de corte que indicam a cadeia genética a ser trocada entre os indivíduos. Na figura 3.5, supõe-se como dois pontos escolhidos para *crossover* 2 e 6. Os pais a serem cruzados são os mesmos do exemplo.

FIGURA 3.5 - *Crossover* com Dois Pontos (MAIA, 2001)



De acordo com a figura 3.5, o material genético foi invertido entre os pais no seguimento anterior e posterior ao seguimento de corte, indicado pelos dois pontos sorteados. Assim os filhos apresentam o código genético mostrado na parte inferior da mesma figura.

No *crossover uniforme* para cada bit no primeiro filho é decidido (com alguma probabilidade fixa  $p$ ) qual pai vai contribuir com seu respectivo bit para aquela posição. Assim, este método não utiliza pontos de cruzamento, mas determina através de um parâmetro global a probabilidade de cada variável ser trocada entre os pais. *Crossover* Uniforme difere também dos demais métodos pois gera apenas um filho de cada par de cromossomos. O processo ocorre conforme mostra a figura 3.6.

FIGURA 3.6 - *Crossover* Uniforme (CONCILIO, 2000)

Pai1	1	0	1	0	0	1	1	1	0	1
Pai2	0	1	1	1	0	1	0	1	1	0
Filho1	1	1	1	1	0	1	1	1	1	1
Filho2	0	0	1	0	0	1	0	1	0	0

O *crossover PMX* é utilizado em problemas onde a ordem com que os genes aparecem no cromossomo dos indivíduos influencia diretamente no valor de *fitness* e na validade do mesmo quanto às restrições impostas pelo problema. Neste método, uma seção



de cruzamento é definida por dois genes aleatoriamente sorteados. Para exemplificar o funcionamento, considera-se os dois indivíduos abaixo:

FIGURA 3.7 - *Crossover* PMX (GOLDBERG, 1989)

A	9	8	4	5	6	7	1	3	2	10
B	9	7	1	2	3	10	9	5	4	6

Primeiro, duas cadeias A' e B' são produzidas a partir da cópia das primeiras. Posteriormente, o processo inicia na cadeia A e mapeia a cadeia B na cadeia A, ou seja: 5 e 2 trocam de posição, o 5 na cadeia A ocupa o lugar de 2 na cadeia A e o 2 na cadeia B ocupa o lugar de 5 na cadeia B até o segundo ponto de cruzamento, assim:

Troca de 5 por 2:

A'	9	8	4	2	6	7	1	3	5	10
B'	9	7	1	5	3	10	9	2	4	6

Troca de 6 por 3:

A'	9	8	4	2	3	7	1	6	5	10
B'	9	7	1	5	6	10	9	2	4	3

Troca de 7 por 10:

A'	9	8	4	2	3	10	1	6	5	7
B'	9	10	1	5	6	7	9	2	4	3

Esse método de cruzamento evita o aparecimento de genes duplicados ou mesmo ausentes no cromossomo dos indivíduos. Desta forma, não é possível produzir indivíduos

inválidos. Além disso, aptidões genéticas conquistadas pelos pais são herdadas pelos filhos.

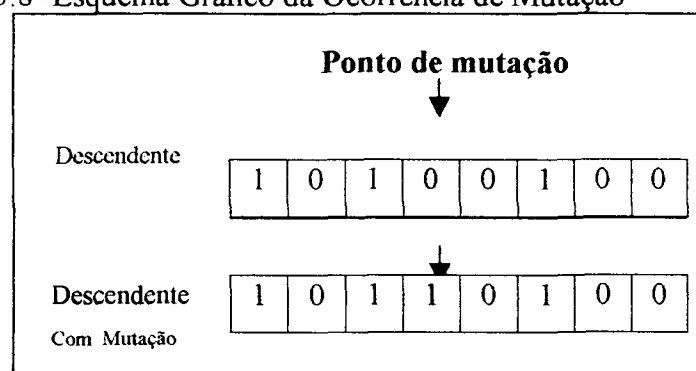
### Operadores de Mutação

Os operadores de mutação são necessários para a introdução e manutenção da diversidade genética da população. A mutação provê meios para a introdução de novos elementos na população promovendo a diversidade e variabilidade extra na população sem atrapalhar o progresso já alcançado pelo algoritmo genético.

O operador de mutação modifica aleatoriamente um ou mais genes de um cromossomo. A probabilidade de ocorrência de mutação em um gene é denominada taxa de mutação. Usualmente, são atribuídos valores pequenos para a taxa de mutação e segundo BACK *et al.* (1997), estes valores podem variar de 0 a 1, sendo que os valores mais utilizados são 0.001 e 0.01.

Considerando uma codificação binária, como o descendente da figura 3.8 sem mutação, o operador de mutação simplesmente troca o valor do gene selecionado, resultando no cromossomo do descendente inferior com mutação, da mesma figura.

FIGURA 3.8 -Esquema Gráfico da Ocorrência de Mutação



### 3.1.7 TEOREMA FUNDAMENTAL DO ALGORITMO GENÉTICO

Para comprovar a validade do funcionamento da abordagem descrita por Algoritmos Genéticos, HOLLAND (1984) elaborou a teoria dos esquemas ou, simplesmente, esquematas. Segundo HOLLAND (1984), “Um esquemata é um modelo de similaridade que descreve um subconjunto de strings com similaridades em determinadas posições da string”.

A hipótese fundamental definida por AG's é que indivíduos com alto grau de *fitness* são portadores de bons esquemas. Esses bons esquemas, por sua vez, podem ser utilizados na construção de indivíduos com aptidões ainda mais altas. O AG constrói, portanto, uma nova geração através da exploração das informações contidas nos blocos de construção dos indivíduos da geração corrente.

Para mostrar melhor o que é um esquemata, utilizar-se-á o exemplo demonstrado em GOLDBERG (1989) através do alfabeto binário  $\{0,1\}$ . É possível produzir um esquemata apenas com o acréscimo de um símbolo especial para este alfabeto: # ou *don't care*, que representa um meta-símbolo, ou seja, um símbolo sobre outros símbolos ou “coringa”. A partir disso, serão criadas strings sobre o alfabeto que agora é ternário  $\{0,1,\#\}$ . Um esquemata, em uma string em particular, consiste da composição dos valores 0, 1 e # sendo que, este último, pode assumir valores 0 ou 1.

Por exemplo, considerando o esquemata e a string de tamanho 5. O esquemata representado por #0000 originará duas strings: {10000, 00000}. Se for considerado o esquemata #111#, o resultado descreverá um subconjunto com quatro membros {01110, 01111, 11110, 11111}. Da mesma maneira, um esquemata 0#1## apresentará como resultado um conjunto de oito strings que começam com 0 e tenham 1 na terceira posição.

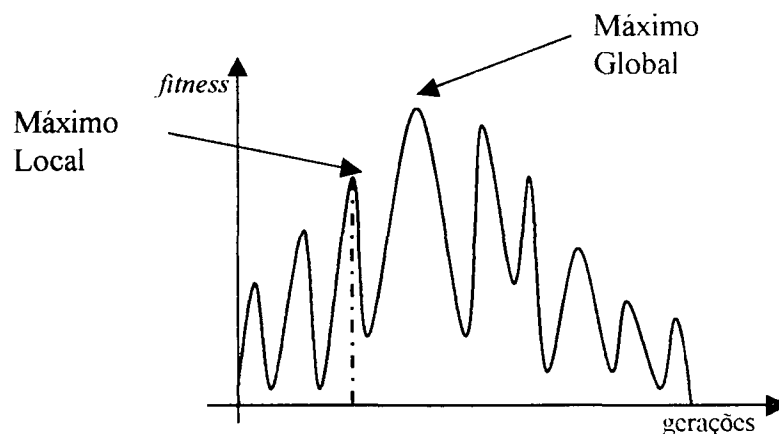
Considerando um esquemata com tamanho  $n = 5$ , ter-se-á  $3 \cdot 3 \cdot 3 \cdot 3 \cdot 3 = 3^5 = 243$  diferentes similaridades, pois cada uma das cinco posições pode assumir valor 0, 1 ou #. Normalmente, num alfabeto com cardinalidade (número de caracteres do alfabeto)  $K$ , ter-se-á  $(K+1)^n$  esquematas.

### 3.1.8 ALGUNS PROBLEMAS ENCONTRADOS EM ALGORITMOS GENÉTICOS

Existem em AG's um fenômeno conhecido como Convergência Prematura. Isto ocorre quando surgem cromossomos de alto *fitness*, mas os cromossomos realmente ótimos ainda não estão presentes na população. Tais cromossomos (chamados de superindivíduos) geram um número excessivo de filhos que dominam a população, uma vez que a mesma é finita, diminuindo conseqüentemente a diversidade da população. Estes cromossomos espalham seus genes por toda a população, enquanto outros genes desaparecem.

Como conseqüência destes acontecimento, o algoritmo converge para um máximo ou mínimo local. A figura 3.9 ilustra este processo.

FIGURA 3.9 - Convergência Prematura dentro do espaço de busca, extraído de NARDIN (1999).



A convergência prematura pode ser combatida limitando-se o número de filhos por cromossomo. Esta limitação pode ser realizada, por exemplo, através do escalamento (*ranking*) do *fitness*. A redução do índice de diversidade da população pode ser combatida aumentando a taxa de probabilidade de mutação e evitando a inserção de filhos duplicados na população.

A função objetivo também é um ponto importante a ser considerado, pois em alguns problemas, pode ser bastante complexa, demandando um alto custo computacional. Existem problemas em que, para avaliar um cromossomo, é necessária uma simulação completa do processo, o que pode chegar a consumir muitas horas. Algumas sugestões são

dadas por CARVALHO e LACERDA (1999) para trabalhar com tais funções objetivo. Estes passos apresentam cuidados a serem tomados para que cromossomos idênticos não sejam avaliados mais de uma vez, aumentando o custo computacional o processo:

- evitar a geração cromossomos idênticos na população inicial;
- verificar se foi aplicado *crossover* ou mutação nos pais, pois, caso não tenha sido aplicado, os filhos são iguais aos pais;
- verificar se o filho é igual a um dos pais;
- manter a população com todos os cromossomos distintos entre si, o que também ajuda na manutenção da diversidade;
- antes de avaliar um filho, verificar se já existe um cromossomo igual a este filho na população. Neste caso, armazenar todos os cromossomos das gerações atuais e passadas, e verificar se algum deles é igual ao novo filho gerado.

É importante esclarecer que estas abordagens também incorporam um custo computacional extra ao AG, por isso, deve-se analisar se este custo extra compensa o tempo economizado na avaliação da função objetivo.

### 3.2 Coevolução Cooperativa

O método da Coevolução Cooperativa foi proposto por POTTER e DE JONG (2000) e aplica a teoria de algoritmos genéticos para diminuir complexidade de problemas, através do conceito de modularização. Esta abordagem pode ser considerada como uma extensão do modelo tradicional de Algoritmos Genéticos, onde é possível representar e solucionar problemas complexos através da modelagem da coevolução de espécies.

Para que a decomposição de problemas e resolução através de algoritmos evolutivos obtenha sucesso, é necessário considerar quatro aspectos importantes ao nível de modularização POTTER e DE JONG (2000):

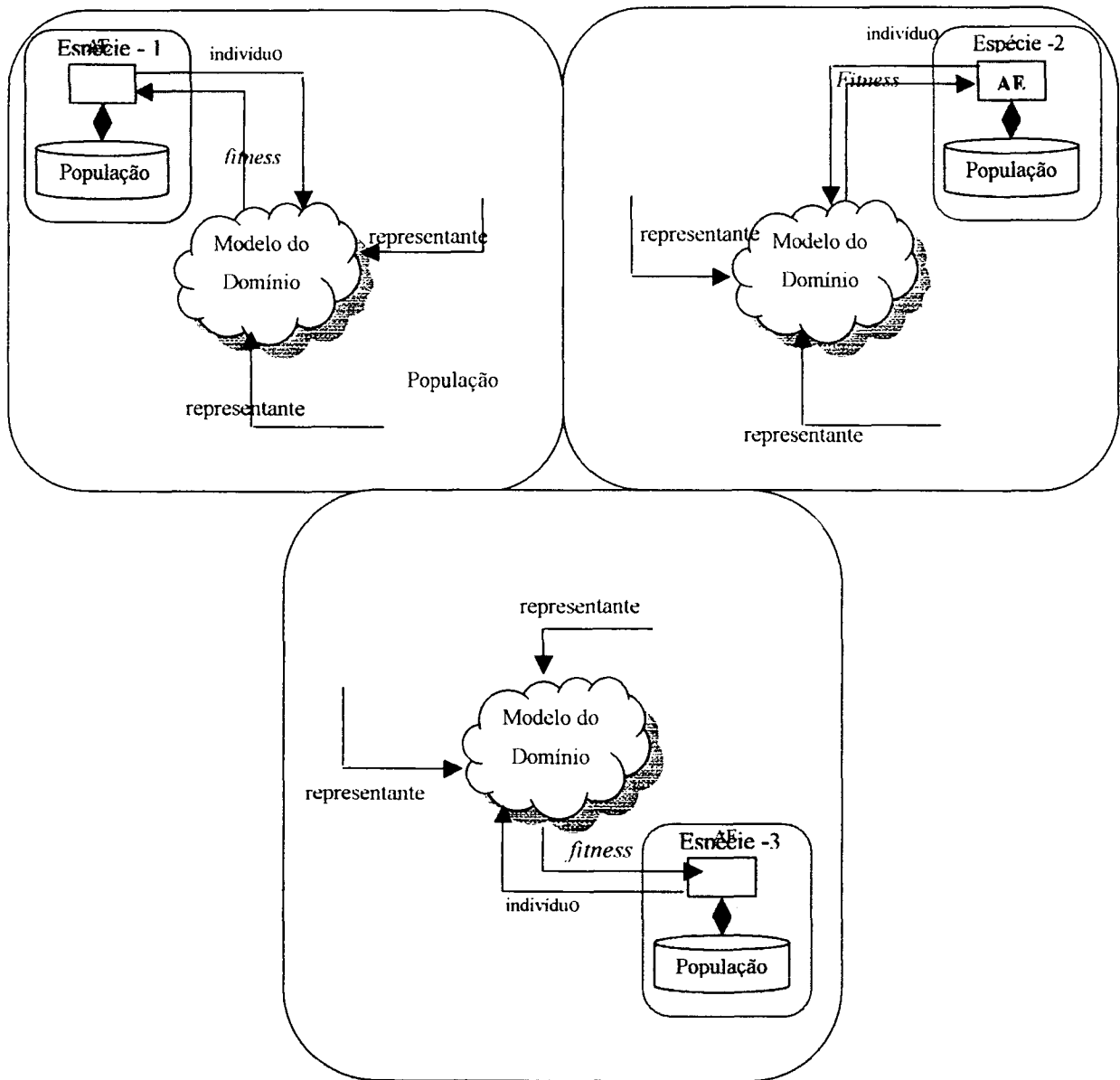
1. **Decomposição:** Determinar um número apropriado de subcomponentes e o papel de cada um destes;
2. **Interdependência entre subcomponentes:** Estabelecer o grau de relacionamento entre os componentes

3. **Atribuição de *fitness*:** Calcular um *fitness* representativo de quanto cada sub-componente está cooperando para a solução do problema.
4. **Manutenção da diversidade:** Em contraste com algoritmos evolutivos, este método utiliza várias populações, que por sua vez possuem características próprias. A solução final é então composta por indivíduos extraídos das diferentes espécies, para compor a solução completa, portanto, não limita a resposta a apenas um indivíduo.

Em essência, o método Coevolutivo define um ecossistema composto de duas ou mais espécies geneticamente isoladas. Assim, um indivíduo pode trocar características somente com membros de sua espécie. As restrições de isolamento durante a evolução das espécies são aplicadas para evitar produção de descendentes inválidos em consequência da troca de material genético por indivíduos altamente especializados. Elas evitam também que indivíduos com valor baixo de *fitness* prejudiquem indivíduos de uma outra espécie.

A figura 3.10 apresenta a arquitetura definida pela Evolução Cooperativa. Nesta figura, a fase que está em evidência é a avaliação do *fitness* da perspectiva de cada uma das espécies envolvidas no ciclo evolutivo.

FIGURA 3.10 - Modelo Coevolutivo Cooperativo para três espécies (POTTER E DE JONG, 2000).



O ciclo evolutivo desenvolvido pelos algoritmos cooperativos está representado no quadro da figura 3.11. Este ciclo parte da inicialização das espécies que compõe o sistema. Se uma solução satisfatória não for encontrada neste momento todas as espécies são evoluídas.

Durante o processo evolutivo, cada espécie é submetida à aplicação dos operadores genéticos de seleção, *crossover* e mutação, descrita anteriormente. As populações antigas são substituídas pelas populações de descendentes, geradas após a aplicação dos operadores.

FIGURA 3.11 - Algoritmo Cooperativo, extraído e adaptado de COSTA e BRUNNA (2000).

```

Algoritmo Coevolutivo Cooperativo

inicio
  Geração=0;
  enquanto (pop<MaxPopulacao) { onde Max População= número máximo de populações
    inicializa_populacao(pop);
    escolhe aleatório representantes[pop]
  }
  enquanto (pop< MaxPopulacao) {
    avalia fitness (pop);
    representantes[pop]=indivíduo com maior fitness;
  }
  enquanto (não (critério de parada)){
    Geracao++;
    enquanto (pop <MaxPopulacao){
      anterior_recebe_atual();
      elitismo();
      selecao();
      mutacao();
      avalia();
    }
    enquanto (pop <MaxPopulacao ) {
      representantes[pop]=indivíduo com maior fitness[pop];
    }
    a cada n gerações {
      avaliar contribuição de cada representante no domínio
      compartilhado do problema();
      inicializa_população improdutiva ()
      avalia o fitness dos indivíduos da população improdutiva reinicializada()
    }
  }
fim

```

Posteriormente, os representantes de cada espécie são selecionados para formar o modelo compartilhado do domínio do problema e viabilizar a avaliação da colaboração das mesmas a cada geração do processo evolutivo.



Para prevenir a estagnação do *fitness* das populações, prejudicando assim o processo evolutivo, a cada  $n$  gerações é executada uma verificação do *fitness* colaborativo. Se alguma das populações estiver com o *fitness* estagnado então seus indivíduos são substituídos por outros através do processo de reinicialização. Esta população é posteriormente avaliada e re-inserida no processo evolutivo.

No capítulo 4 os processos abrangidos no algoritmo apresentado na figura 3.11 são abordados em maiores detalhes. A Evolução Cooperativa é o método adotado por este trabalho para solução de *timetabling*.

### 3.2.1 AVALIAÇÃO DO *FITNESS* E ESCOLHA DE REPRESENTANTES

Algoritmos Cooperativos efetuam atribuição de *fitness* relativo ao nível de colaboração de uma espécie no ambiente que está inserida. Para que esse valor possa ser calculado, representantes de diferentes espécies devem ser escolhidos para formar o conjunto domínio do problema, e uma função de *fitness* é aplicada sobre os indivíduos.

Existem diversos meios de se escolher representantes que irão compor o modelo de domínio compartilhado do problema. Segundo POTTER e DE JONG (2000) em alguns casos, é apropriado escolher como representante simplesmente o indivíduo com o melhor *fitness* atual de cada uma das espécies. Em outros casos esta escolha pode ser feita de forma aleatória.

### 3.2.2 ELIMINAÇÃO DE POPULAÇÕES IMPRODUTIVAS

Durante o ciclo cooperativo, espera-se que a evolução ocorra de maneira uniforme em todas as populações promovendo uma melhora gradativa no *fitness* de seus indivíduos. Esta população é, portanto denominada produtiva. Porém, segundo POTTER e DE JONG (2000), um problema que pode ocorrer, durante a evolução das espécies em algoritmos coevolutivos cooperativos, é a estagnação dos valores de *fitness* apresentados por uma população ou espécie dita improdutiva.

Para que o processo evolutivo possa atingir com sucesso o seu objetivo, ou seja obtenção de uma solução ótima através da melhora gradativa dos indivíduos, a abordagem cooperativa efetua a reinicialização das espécies improdutivas. O processo utilizado é o

mesmo da inicialização das populações, sendo somente as estagnadas submetidas a este processo.

Depois da reinicialização, os novos indivíduos são reintegrados no processo evolutivo e todos os processos normais pertinentes à evolução são executados normalmente.

Segundo POTTER e DE JONG (2000), a estagnação da evolução pode ser detectada através da monitoração da qualidade das colaborações através da aplicação da função:

$$f(t)-f(t-K)<C,$$

onde  $f(t)$  é considerado o valor de melhor *fitness* de colaboração em um tempo  $t$ ,  $C$  é uma constante especificando o aumento no valor de *fitness*, e  $K$  é uma constante especificando o tamanho de uma janela evolutiva na qual significativas melhoras podem ser realizadas.

### 3.3 Esquemas Híbridos

Quando os elementos de algum problema específico são bem conhecidos, a utilização de esquemas híbridos é vantajosa. Algoritmos Genéticos podem ser empregados juntamente com técnicas de busca local permitindo uma exploração mais abrangente no espaço de soluções e conseqüentemente localização de soluções melhores.

Segundo GOLDBERG (1989), existem diversas abordagens que exploram o conceito de Esquema Híbrido. Em problemas de otimização numérica sujeitos a restrições, onde o método de AG's pode ser empregado com sucesso, um esquema híbrido pode ser descrito como um procedimento de busca local empregado a partir da solução dada pelo Algoritmo Genético. O processo busca soluções melhores na vizinhança da hipótese candidata dada. Neste caso pode então ser aplicado o método de *Hill Climbing*. O resultado hibridização é a melhora da qualidade das soluções, ou seja, adaptação destas às restrições do contexto do problema.

### 3.4 Algoritmos Meméticos

Segundo CONCÍLIO (2000), cada indivíduo coleta ao longo de sua vida uma grande quantidade de informações. Essa massa de dados, em conjunto com observações e conclusões próprias contribuem para a formação de idéias sobre diversos assuntos. Pode-se dizer que as idéias formadas ao longo da vida humana, são expostas a um processo de “seleção” ou refinamento através do confronto entre as conclusões obtidas, fatos observados e dados coletados.

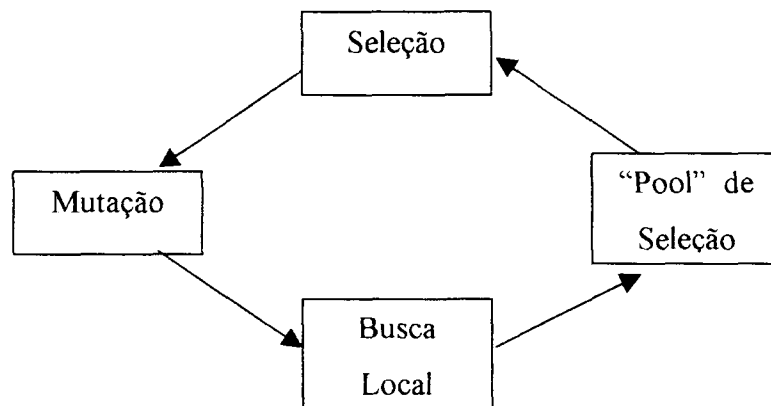
Neste processo de seleção, as idéias incoerentes são descartadas, e as demais são mantidas. Este conjunto pode ser visto como uma população de idéias que ao longo do tempo e através do conhecimento adquirido, produz melhoria na capacidade intelectual desenvolvida pelos seres vivos.

Segundo CONCÍLIO (2000, p.31), o processo de busca por melhores respostas somente pode ser concebido dentro dos limites do campo do conhecimento, na forma de um processo iterativo aplicado por um indivíduo para a melhoria contínua de suas idéias. No campo da computação evolutiva, o refinamento do conhecimento pode ser incorporado como uma etapa de apoio ao processo evolutivo.

O conceito de Algoritmos Meméticos foi primeiramente abordado por MOSCATO e NORMAM (1991) para descrever processos evolutivos que possuam uma busca local como parte decisiva no processo de evolução. Essa busca pode ser caracterizada como sendo um refinamento local dentro de um espaço de busca. O termo *meme* foi idealizado por DAWKINS (1976) como sendo uma unidade de informação que se reproduz durante um processo argumentativo e de transmissão de conhecimento. De acordo com CONCÍLIO (2000, p.32) quando o meme é transmitido ele será adaptado pela entidade que o recebe com base no seu conhecimento e para melhor atender suas necessidades.

A idéia geral dos Algoritmos Meméticos é a utilização dos operadores evolutivos que determinam regiões promissoras no espaço de busca combinados com busca local nestas regiões. Assim, algoritmos meméticos correspondem à união de um método de busca global e uma heurística local aplicada a cada indivíduo, de modo que um algoritmo memético é na verdade um tipo especial de busca local. O ciclo desenvolvido pelos algoritmos meméticos está representado na figura 3.12.

FIGURA 3.12 - Ciclo evolutivo Memético (NEWALL, 2000).

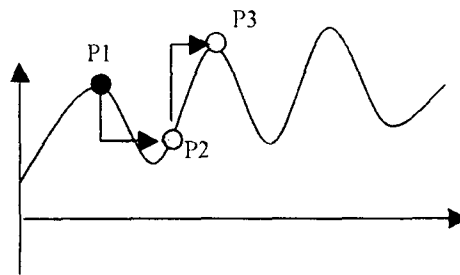


Na figura 3.12, sobre uma população inicial, é aplicado o processo de seleção. Os indivíduos selecionados passam pelo processo de mutação, e posteriormente segue-se a aplicação do procedimento de busca local.

Conforme CONCÍLIO (2000), durante o processo de geração da população inicial, se alguma restrição à que o problema está sujeito for violada, indivíduos inválidos são produzidos. Existe portanto a necessidade da utilização de algum dispositivo seletivo capaz de discernir entre indivíduos factíveis ou não. Além disso, algoritmos de reparação também podem ser utilizados como uma boa alternativa para factibilização de indivíduos inválidos, através da readaptação de alguns memes.

Segundo NEWALL (2000), “a principal vantagem ganha pelo uso de Algoritmos Meméticos é a redução do espaço de possíveis soluções para o sub-espaço do ótimo local”. Um exemplo do resultado obtido com o emprego deste método é representado na figura 3.13.

FIGURA 3.13 – Adição de busca local a ciclos evolutivos (NEWALL, 2000)



No exemplo da figura 3.13, o P1 representa um indivíduo que se localiza em uma região do espaço de busca que contém um máximo local. Após a mutação observa-se que ele foi remetido para fora do espaço original do ótimo local (P2). Quando uma busca local é aplicada ao indivíduo levou o descendente a assumir um *fitness* elevado, promovendo a localização do indivíduo P3 no espaço do máximo global. Os Algoritmos Meméticos serão novamente abordados no capítulo 4.

É possível observar através dos diferentes métodos abordados neste capítulo que, devido ao grande leque de recursos oferecidos, a abordagem evolutiva pode ser considerada como uma ferramenta poderosa. Além disso, a abordagem provê um arcabouço de idéias que promove o desenvolvimento e aprimoramento contínuo de novas técnicas e algoritmos evolutivos. Assim, problemas considerados complexos podem ser resolvidos de forma mais eficiente do que através em métodos convencionais.

No próximo capítulo são apresentados métodos não evolutivos, que também já foram utilizados para a resolução de problemas de escalonamento como *timetabling*.

## 4 TRABALHOS RELACIONADOS

Para a resolução o problema (*Timetabling*) que está sendo analisado neste trabalho, outras técnicas já foram utilizados anteriormente. Nesta seção, serão vistos os principais aspectos de técnicas como Heurísticas Dirigidas, Grafos Coloridos, Programação Lógica e AG's com penalidades.

### 4.1 Heurísticas Dirigidas

Para se chegar a uma *timetable* completa a abordagem de Heurísticas Dirigidas promove alocação e movimentação de atividades dentro das tabelas até que todas as atividades pertinentes ao problema estejam devidamente alocadas não provocando quaisquer conflitos entre horários. Um típico exemplo deste método é o sistema SCHOLA apresentado por Uhleman apud SCHAERF(1995). Este sistema baseia-se nas seguintes estratégias:

- a) atribuir atividades mais urgentes nos periodos mais favoráveis para tais atividades;
- b) quando um período pode ser usado somente para uma atividade, atribuir a este período tal atividade.
- c) mover uma atividade já escalonada para outro período livre quando se precisa liberar o período para a atividade que está se tentando escalonar atualmente.

Para que uma atividade seja considerada “urgente” ela deve estar relacionada com um grande número de restrições, ou seja, ter poucos recursos disponíveis para que sua alocação na *timetable* seja efetivada. Para que um periodo seja considerado “favorável” a uma atividade, os recursos disponíveis para que outras atividades sejam escalonadas neste, são escassos.

O sistema SCHOLA escalona atividades alternando entre as estratégias A e B até enquanto isto for possível. Quando as atividades não puderem mais ser escalonadas desta forma, a estratégia a ser adotada é a C.

## 4.2 Grafos Coloridos

Neufeld e Tartar *apud* SCHAERF (1995) propuseram um método para a resolução de *timetable* através de grafos coloridos. Neste método, cada atividade é associada a um vértice no grafo e entre cada par de atividades que não podem ser escalonadas ao mesmo tempo existe um arco. Por exemplo, duas atividades que compartilham o mesmo professor e a mesma sala (ou ambos) são ligadas pelo arco.

As restrições inerentes ao problema que está sendo resolvido, são expressas no grafo através da coloração de vértices específicos. O grafo resultante e todas as suas restrições podem facilmente tornar-se uma *timetable* fazendo-se a seguinte associação: a cada período da tabela é atribuída uma das cores existentes no grafo.

Posteriormente todas as atividades pertinentes a um vértice devem ser escalonadas em um período na *timetable* que corresponde a sua cor no grafo.

## 4.3 Programação Lógica

KANG e WHITE (1992), usaram PROLOG para implementar uma aplicação computacional para *timetabling*. A principal vantagem desta abordagem é a flexibilidade em expressão de uma forma declarativa das restrições envolvidas no problema. Além disso, recursos como o *backtracking* incluso na máquina PROLOG permitem o reescalonamento de colisões ou conflitos criados.

Em particular quando uma atividade não puder ser inserida em determinado período, o processo busca uma atividade equivalente já escalonada e a atribui a outro período diferente. Caso não existam atividades equivalentes que possam ser reescalonadas, uma lista de atividades, não escalonáveis, é gerada para que uma posterior escala manual da *timetable* possa ser realizada.

#### 4.4 Algoritmos Genéticos com penalidades

Os Algoritmos Genéticos podem, durante o processo de avaliação dos indivíduos, aplicar técnicas conhecidas como funções de penalidade. Segundo OLIVEIRA (2001) estas funções empregam pesos diferentes para cada restrição violada. Assim, quanto mais restrições forem violadas, mais baixo será o *fitness* do indivíduo.

Os pesos utilizados, servem para ponderar as penalidades e definir quais são as mais importantes para o processo de otimização. Portanto, é importante que os pesos estejam corretamente ajustados para dar um comportamento suave a cada uma das restrições da função objetivo modificada.

Para o emprego de funções de penalidade existem duas estratégias, conforme OLIVEIRA (2001): penalidades uniformes ou variáveis. Penalidades uniformes, apresentam pesos fixos ao longo do processo evolutivo. Já as penalidades variáveis alteram gradativamente seu valor, de forma que no início do processo os mesmos são aplicados a uma “pressão menor” que é gradativamente aumentada no decorrer do processo.

O objetivo do emprego de penalidades variáveis na função objetivo, consiste em aumentar o número de soluções candidatas factíveis no espaço de busca. A medida que a população evolui, as soluções infactíveis são progressivamente penalizadas reduzindo assim o espaço de busca.

Nas seções seguintes apresenta-se dois algoritmos evolutivos para a solução de *timetabling*. O primeiro método é conhecido como Algoritmo Memético, foi desenvolvido por NEWALL (2000) e foi aplicado ao contexto de produção de *timetables* em universidades. Posteriormente, é apresentado o algoritmo Coevolutivo, implementado para este trabalho. Este método foi aplicado na resolução do problema de geração de grades horárias para o curso de Bacharelado em Ciência da Computação da Universidade Federal do Paraná.



## 4.5 Algoritmos Meméticos

O ciclo descrito por algoritmos meméticos quando aplicados a *timetabling*, é idêntico ao visto no capítulo anterior. Neste ciclo, o processo de busca local é utilizado juntamente com operadores meméticos procurando remeter o descendente gerado a um ótimo global. É apresentada a seguir a solução proposta por NEWALL (2000) para o problema *timetabling* a partir da abordagem de Algoritmos Meméticos.

### 4.5.1 REPRESENTAÇÃO DA SOLUÇÃO PARA *TIMETABLING* DE EXAMES.

Segundo NEWALL (2000), a solução proposta através Algoritmos Meméticos para *timetable*, pode ser codificada de forma direta, armazenando a tabela literalmente, ou indireta, codificando instruções de como construí-la.

A representação da solução é feita por um determinado número de memes, e é apresentada na tabela 4.1. Cada meme contém informações sobre o escalonamento de salas e eventos para um determinado período. Um dos memes do indivíduo é utilizado para armazenar provas que não podem ser escalonados nos períodos prescritos.

TABELA 4.1 - Codificação da solução para *timetable* (BURKE e NEWALL, 1996).

Período 1		...	Período n		Não escalonados
Sala 1	e1,e2,e4		Sala 1	e5,e22,e24	e53,39
Sala 2	e3 e e7		Sala 2	e37, e57	
Sala 3	e11, e36		Sala 3	e45	
Sala 4	e90 e e27		Sala 4	e63,e52	

### 4.5.2 GERAÇÃO DA POPULAÇÃO INICIAL.

NEWALL (2000), explica que a inicialização das populações seleciona um evento aleatoriamente e atribui este evento ao período que mais favorecer esta inserção. A verificação dos períodos mais favoráveis se dá através de um cálculo de uma função heurística. Assim a heurística é utilizada juntamente com a aleatoriedade no processo de inicialização para garantir a qualidade e a variabilidade das soluções.

A rotina básica para gerar indivíduos para a população inicial é resumido por NEWALL (2000), como:

1. remover um evento  $e$  aleatório da lista de atividades. Se todos exames tiverem sido escalonados então finalizar.
2. para cada período na *timetable*, determinar se é legal inserir  $e$  no período em questão. Se sim, calcular uma medida de quão bom será seu uso através da função :  $((numComum + Tamanho + 1) / penalidade + 1)$ , onde  $numComum$  é o número de fronteiras que  $e$  tem em comum com exames já escalonados no período e  $penalidade$  é o custo da inserção no período. Esta função heurística primeiro escalona eventos que tem conjuntos similares de fronteiras e depois tenta minimizar a penalidade causada pelo escalonamento de  $e$  através da indução da Roleta nos períodos que favorecem  $e$  com menor penalidade.
3. construir e executar uma Roleta baseada nos valores calculados para escolher um período para inserir  $e$ .

#### 4.5.3 AVALIAÇÃO DOS INDIVÍDUOS

As restrições apresentadas por BURKE e NEWALL (1996) para a produção de *timetables* válidas, no contexto de escala de exame em universidades, podem ser descritas como:

- nenhum estudante pode ter dois exames ao mesmo tempo;
- se um estudante tem dois exames no mesmo dia, então deve existir um período entre os dois exames;
- o limite máximo de lugares para exames é 1550 no contexto do problema analisado, e não pode ser excedido.

De acordo com as restrições anteriores, a função de avaliação foi definida como:

$$200 \frac{10 \text{ num } Eventos}{\text{Não escalonado} + \sum_{i=0}^{\text{Num } Períodos - 1} \text{Num } Conflitos(\text{período } i, \text{período } i+1))}$$

onde *numEventos* é o número de eventos na *timetable*, *numNãoescalonado* é o número de eventos não escalonados em períodos ainda, e *NumConflitos* é a função que retorna o número de conflitos, multiplicada pelo número de estudantes entre dois períodos no mesmo dia. Pode-se concluir assim que a avaliação da qualidade das *timetables* é calculada sobre as restrições por ela violadas.

#### 4.5.4 OPERADORES MEMÉTICOS E BUSCA LOCAL

O Algoritmo Memético desenvolvido por NEWALL (2000), aplica uma combinação de operadores de mutação (*light* e *heavy*) e operador de seleção. Esses processos são imediatamente seguidos pela aplicação de um método de busca local.

##### 4.5.4.1 OPERADOR *LIGHT* DE MUTAÇÃO

Segundo NEWALL (2000), operador *light* de mutação escolhe um número de eventos aleatórios de qualquer ponto do indivíduo e executa sua re-inserção em qualquer outro período legal sendo seguido da aplicação da busca local.

##### 4.5.4.2 OPERADOR *HEAVY* DE MUTAÇÃO

O operador *heavy* de mutação, é usado para dividir ou “quebrar” um ou mais conjuntos de períodos em um indivíduo. Neste operador períodos que não violam restrições do problema são preservados e os demais períodos são utilizados para re-escalonamento de eventos, produzindo assim novas soluções de alta qualidade.

O processo que permite determinar se um dado período deve ser preservado ou não, é resumido como:

- tomar cada período *i* na *timetable* em questão.
  - Calcular a *penalidade* resultante dos eventos no período *i* assumindo que o evento escalonado no período *j* (onde  $j > i$ ) deve permanecer fixo;

- Se esta penalidade é menor que a penalidade média na população, então verificar se o período será rompido de acordo com uma probabilidade pré-estabelecida
- Se a decisão é romper então
  - Se o prévio período foi rompido então romper o **próximo** período
  - Senão, romper **este** período.

Segundo NEWALL (2000), quando forem escolhidos quais os períodos serão rompidos, os exames pertinentes são agrupados e cada exame deste grupo é randomicamente re-escalonado no primeiro período legal. Esta operação só é válida quando for seguida da aplicação da busca local para atingir uma melhora substancial através de um novo local ótimo.

#### 4.5.5 BUSCA LOCAL

Segundo NEWALL (2000), a busca local é utilizada para promover melhorias na qualidade do *fitness* dos indivíduos e envolve basicamente o cálculo da variação do *fitness* quando acontece uma movimentação de atividades. A vantagem da utilização deste método, reside na redução de tempo consumido para avaliações completas apresentadas por outros métodos.

O procedimento de busca local utilizado para a abordagem Memética está resumido em NEWALL (2000) como:

- Tomar cada período
  - Tomar cada evento  $e$  neste período
    - \* Para cada período  $p$  na *timetable*, calcular a penalidade que resultará do escalonamento do evento  $e$  no período  $p$  se restrições *soft* forem violadas;
    - \* Escalonar  $e$  no período que possui menor penalidade.
  - Tentar escalonar os eventos da lista de eventos não escalonados.

Neste capítulo foram abordados métodos para solução de *timetabling* que trabalham com soluções candidatas de maneira diferente que na computação evolutiva. No próximo capítulo é apresentado um estudo de caso sobre a Geração de Grades Horárias em uma Universidade, bem como a forma com a qual Algoritmos Cooperativos e Algoritmos Genéticos foram utilizados para o desenvolvimento de sistemas que se propõe a resolver o problema.

## 5 ESTUDO DE CASO

Neste capítulo são apresentadas as principais características relacionadas ao problema em análise, bem como sua modelagem. Também são apresentadas as duas abordagens, Evolução Cooperativa e Algoritmos Genéticos, implementadas para a resolução deste problema.

### 5.1 Abordagem Cooperativa para o problema *Timetabling*

Nesta seção, procurou-se implementar e verificar a validade da Evolução Cooperativa para a resolução de *timetabling*, no contexto abordado de Geração de grades Horárias para instituições de ensino.

A evolução Cooperativa foi o método escolhido pois; seus aspectos conceituais favorecem a modelagem de problemas complexos como *timetabling* através da divisão deste em problemas menores e, logicamente, menos complexos. Além disso, como esta abordagem é uma extensão de Algoritmos Genéticos, pode ser aplicada a problemas de otimização e até o momento foi pouco explorada.

#### 5.1.1 MODELAGEM DO PROBLEMA

O estudo de caso analisado é o curso de Ciência da Computação da Universidade Federal do Paraná. Este curso tem uma duração de 4 anos dividido em 8 períodos semestrais. Cada período ou turma tem cinco dias de aulas, representando uma semana, e cada dia está distribuído em três horários que devem ser ocupados com disciplinas. O quadro atual de professores conta com 33 profissionais e a grade curricular total dos períodos é composta por 50 disciplinas.

Como a arquitetura da evolução cooperativa define, o problema original deve ser dividido em subproblemas a serem representados pelas diferentes espécies componentes

do sistema. Desta forma, para o problema escolhido, a modularização ocorre de forma natural uma vez que o curso é dividido em oito períodos. Assim, o sistema a ser evoluído é composto, por oito espécies. A representação dos indivíduos das espécies será vista posteriormente.

Ainda de acordo com a arquitetura, a evolução cooperativa do sistema se dará quando todas as turmas apresentarem escalas compatíveis com as restrições impostas pela Grade Curricular do curso e demais aspectos envolvidos no contexto do problema. Ao final do processo, representantes escolhidos de cada uma das populações serão componentes de uma solução completa para o problema inicialmente decomposto.

### 5.1.2 RESTRIÇÕES

Como *timetabling* consiste da otimização numérica da função objetivo, sendo que o espaço de soluções factíveis está sujeito à restrições, definiu-se para o contexto apresentado, as seguintes restrições:

1. *Disciplina\_Professor*: Essa restrição dita que disciplinas podem ser ministradas somente por professores pré-determinados.
2. *Soma\_de\_Aulas*: Cada disciplina ministrada tem um número de carga horária semanal que deve ser cumprido.
3. *Disciplina\_Turma*: A cada período ou turma na instituição de ensino somente podem ser ministradas disciplinas pertinentes ao contexto da turma.
4. *Professor\_horário*: Esta restrição diz respeito à preferência ou disponibilidade de horários de trabalho por parte dos professores. Para modelagem do problema as preferências de horários foram expressas em termos numéricos que variam de 1 a 5. Estes números são proporcionais ao grau de indisponibilidade de determinado professor, assim, quando maior o número, menor é a preferência do professor por determinado horário.
5. *Limite de horas/aula por professores*: Cada professor tem limite máximo semanal de carga horária estabelecido em doze aulas e o mínimo em duas aulas.
6. *Conflito de horários*: Nenhum professor pode ser alocado em dois horários idênticos em turmas diferentes.

7. Aulas Geminadas: Quando a mesma disciplina é ministrada em um mesmo dia, duas ou mais vezes seguidas, tem-se a situação conhecida como aula geminada. No sistema implementado, procurou-se evitar que esta ocorresse, constituindo portanto uma restrição que penaliza o *fitness* dos indivíduos.

### 5.1.3 REPRESENTAÇÃO DOS INDIVÍDUOS

Os indivíduos que compõem cada uma das espécies na evolução cooperativa sempre apresentam peculiaridades inerentes à sua espécie. Para o estudo de caso, estas peculiaridades são definidas pelos conjuntos de professores e disciplinas.

Para cada indivíduo inserido na população tem-se como conjunto de genes (cromossomo) a representação de um período. O número de genes de cada indivíduo é de 15 para representar os 5 dias da semana e os três horários de aula.

Cada alelo do indivíduo é instanciado com um disciplina e um professor inerente ao período em questão. A representação adotada está representada na Figura 5.1.

FIGURA 5.1- Representação do cromossomo.

	Segunda	Terça	Quarta	Quinta	Sexta
1º horário	Disc A Prof 1	Disc B Prof 2	Disc C Prof 3	Disc D Prof 4	Disc E Prof 5
2º horário	Disc F Prof 6	Disc C Prof 3	Disc B Prof 2	Disc G Prof 7	Disc H Prof 8
3º horário	Disc A Prof 1	Disc D Prof 4	Disc F Prof 6	Disc H Prof 8	Disc G Prof 7

### 5.1.4 INICIALIZAÇÃO DE POPULAÇÕES

No algoritmo implementado, o primeiro processo a ser executado é a geração da população inicial ou geração zero. A inicialização consiste da instanciação de todos os indivíduos para cada uma das oito espécies com disciplinas e professores correspondentes.



Conforme citado anteriormente é importante, no contexto de problemas sujeitos a restrições, que na primeira geração, indivíduos de alta qualidade genética sejam gerados, ou seja, sejam criados de acordo com as restrições do problema. Este cuidado permite que uma melhor desempenho no processo evolutivo seja obtido.

A principal característica apresentada pelo procedimento de inicialização das populações é a validação das restrições de carga horária semanal de disciplinas e professores além da não existência de colisão de horários de professores para os diferentes períodos. Ao final da geração da população inicial, estas restrições estão satisfeitas, entretanto no decorrer do processamento são constantemente verificadas, já que com a aplicação dos operadores genéticos as grades horárias são modificadas.

Desta forma, o algoritmo executa, antes de cada gene ser instanciado, uma verificação junto às duas restrições descritas acima. Se esta verificação denunciar a violação de alguma delas em conjunto com os representantes das demais espécies, outro valor deverá ser sorteado até que resulte em um indivíduo válido. O processo se repete até que todas as populações estejam completas.

No desenvolvimento deste trabalho, a preferência de professores por horários e colisão de horários entre turmas, não foram verificadas no momento da inicialização das espécies pois estas restrições consistem no objetivo da otimização durante o processo evolutivo e são vislumbradas na função de *fitness* (juntamente com as demais). Em outras palavras, caso o processo de geração de indivíduos iniciais impusesse que, somente os professores com horários disponibilizados com preferência 1 fossem alocados, o papel do ciclo evolutivo seria anulado. Assim cabem aos demais processos, componentes do algoritmo cooperativo, evoluir as populações e encontrar durante processo de busca a solução mais adequada a esta restrição.

#### 5.1.5 AVALIAÇÃO DE *FITNESS*

A função adotada para o cálculo do *fitness* implementada no algoritmo pode ser descrita como:

$$\text{fitness do indivíduo } i = 1/(1 + (\text{restrições\_violadas}))$$

Como durante todo o ciclo evolutivo procura-se minimizar o número de restrições violadas pelos indivíduos, o valor máximo de *fitness* obtido será de 1.0 somente no caso em que a variável “restrições\_violadas” seja nula. Esta variável compreende o número total de restrições “feridas” pelo indivíduo em avaliação no domínio compartilhado do problema. Toda vez que uma restrição pertinente ao problema é violada, a variável *restrições\_violadas* é incrementada em uma unidade.

Pela sua importância, a restrição de preferência de horário do professor recebeu peso diferenciado das demais, caso o professor alocado tenha indicado um valor diferente de 1 para o horário (gene) em avaliação. Assim, toda vez que um indivíduo possui uma instância diferente da ideal para preferência de professor, o valor identificado neste horário é multiplicado por 10 e somado ao número de restrições violadas pelo indivíduo fazendo com que o *fitness* do referido cromossomo seja baixo. Pode-se concluir a partir da definição de pesos mais restritos para os valores de preferência, que a função de *fitness* privilegia e tenta melhorar os indivíduos durante o ciclo evolutivo para que apresentem valor 1 de preferência para todos os professores.

O valor de *fitness* para cada indivíduo é, portanto, inversamente proporcional ao número de restrições por ele violadas, ou seja, quanto maior o número de restrições violadas, menor será o valor resultante do *fitness*. Os indivíduos com *fitness* baixo vão sendo gradativamente substituídos por indivíduos melhor adaptados. A avaliação da aptidão dos indivíduos é um componente de suma importância no processo evolutivo das espécies.

#### 5.1.6 ESCOLHA DOS REPRESENTANTES

A forma de escolha dos representantes pode variar de acordo com a natureza de cada problema. É simples perceber que a função definida para avaliação de *fitness* resulta em valores maiores somente para indivíduos que violaram um número menor de restrições. Assim, para o estudo de caso, adotou-se o método direto de escolha dos representantes os indivíduos com valor mais alto de *fitness* de cada espécie.

### 5.1.7 ELITISMO

O procedimento de elitismo garante a permanência de indivíduos com valores elevados de *fitness* em gerações posteriores durante o ciclo evolutivo. Este processo toma um número  $n$  de indivíduos, pertencentes a geração  $t$  que possuem melhor *fitness* e faz uma cópia destes em uma geração  $(t+1)$ . O valor  $n$  equivale à porcentagem de indivíduos na população a serem mantidos durante as gerações até que indivíduos de *fitness* mais alto venham a substituí-los.

### 5.1.8 OPERADORES GENÉTICOS

O operador de seleção implementado escolhe indivíduos pelo método Roleta, descrito anteriormente. A este processo segue-se a verificação da probabilidade da aplicação do operador genético de *crossover* para os dois indivíduos. Se eles forem sujeitos ao cruzamento de acordo com a probabilidade anteriormente citada, então o processo de *crossover* é iniciado, caso contrário, os indivíduos são copiados para a próxima geração. O *crossover* adotado foi o método PMX abordado anteriormente.

Para representar o funcionamento do *crossover* adotado, considerou-se que a turma zero possui 6 disciplinas, codificadas de 0 a 5 e somente 12 aulas semanais em virtude da carga horária das referidas disciplinas, resultando no exemplo de tabela horária na figura 5.2 abaixo. De acordo com esta configuração, supondo dois pontos de corte sorteados (um e três) que definem a seção de cruzamento hachurada, temos a seguinte representação:

FIGURA 5.2 - *Crossover* implementado.

	Segunda	Terça	Quarta	Quinta	Sexta	<b>Pai 1</b>
<b>1º horário</b>	Disc 0 Prof 0	Disc 4 Prof 3	Disc 0 Prof 0	Disc 5 Prof 4	-	
<b>2º horário</b>	Disc 2 Prof 2	Disc 2 Prof 2	Disc 1 Prof 1	Disc 1 Prof 1	-	
<b>3º horário</b>	Disc 3 Prof 0	Disc 3 Prof 0	Disc 4 Prof 3	Disc 5 Prof 4	-	

	Segunda	Terça	Quarta	Quinta	Sexta	<b>Pai 2</b>
<b>1º horário</b>	Disc 0	Disc 0	Disc 4	Disc 5	-	
	Prof 0	Prof 0	Prof 3	Prof 4		
<b>2º horário</b>	Disc 1	Disc 2	Disc 1	Disc 3	-	
	Prof 1	Prof 2	Prof 1	Prof 0		
<b>3º horário</b>	Disc 5	Disc 3	Disc 2	Disc 4	-	
	Prof 4	Prof 0	Prof 2	Prof 3		

De acordo com o *crossover* PMX, primeiro copia-se o cromossomo do Pai1 para o Filho1 e o cromossomo do Pai2 para Filho2. Posteriormente, mapeando o Pai1 em Pai2 teremos as seguintes trocas para os filhos produzidos: Disciplina 2 por disciplina 0, disciplina 3 por disciplina 2 e disciplina 0 por disciplina 3. Estas trocas resultam nos seguintes indivíduos:

	Segunda	Terça	Quarta	Quinta	Sexta	<b>Filho 1</b>
<b>1º horário</b>	Disc 4	Disc 0	Disc 4	Disc 3		
	Prof 3	Prof 0	Prof 3	Prof 0	-	
<b>2º horário</b>	Disc 1	Disc 1	Disc 2	Disc 2		
	Prof 1	Prof 1	Prof 2	Prof 2	-	
<b>3º horário</b>	Disc 5	Disc 5	Disc 0	Disc 3		
	Prof 4	Prof 4	Prof 0	Prof 0	-	

	Segunda	Terça	Quarta	Quinta	Sexta	<b>Filho 2</b>
<b>1º horário</b>	Disc 4	Disc 4	Disc 0	Disc 3	-	
	Prof 3	Prof 3	Prof 0	Prof 0		
<b>2º horário</b>	Disc 2	Disc 1	Disc 2	Disc 5	-	
	Prof 2	Prof 1	Prof 2	Prof 4		
<b>3º horário</b>	Disc 3	Disc 5	Disc 1	Disc 0	-	
	Prof 0	Prof 4	Prof 1	Prof 0		

O operador de mutação é aplicado somente aos indivíduos sujeitos a uma taxa pré-definida. Caso a mutação ocorra, dentre os genes de um mesmo indivíduo são

aleatoriamente sorteadas duas posições. A instância inserida nestes genes é trocada e o *fitness* do indivíduo é recalculado.

#### 5.1.9 COLABORAÇÃO E ESTAGNAÇÃO ENTRE POPULAÇÕES

Uma tarefa pertinente ao processo de um algoritmo cooperativo é avaliar quanto cada representante de cada população está cooperando para o conjunto colaboração. O valor que expressa a colaboração entre as populações do sistema é denominado *fitness colaborativo*. No algoritmo implementado, esse valor atinge o valor máximo de 1 quando os representantes das oito populações não violarem quaisquer restrições impostas pelo problema.

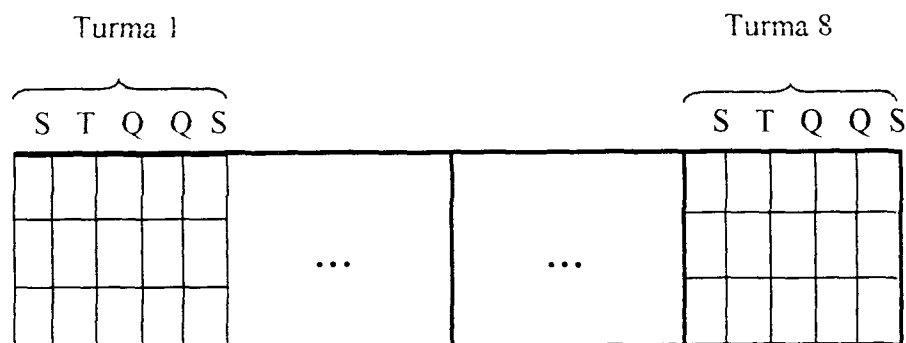
Durante a execução do algoritmo Coevolutivo, a avaliação da contribuição dos representantes para o conjunto colaboração ou domínio compartilhado ocorre esporadicamente, em um número pré-determinado de vezes. Para verificar a contribuição das populações para o conjunto, é avaliada a condição de estagnação do *fitness*. Assim, se em um número pré-determinado de gerações uma mesma população não apresentou nenhuma evolução do valor de seu *fitness* ela é definida como estagnada. A população que apresentar estado de estagnação, será reinicializada e novos indivíduos serão inseridos na população.

### 5.2 Abordagem Genética para o problema *Timetabling*

Para que um comparativo entre Algoritmos Cooperativos e Algoritmos Genéticos fosse viabilizado, foi também implementado um algoritmo Genético na sua versão clássica.

Como o ciclo básico desenvolvido pelo Algoritmo Genético já foi amplamente discutido neste trabalho, esta seção limita-se as especificidades do algoritmo implementado. A modelagem adotada para os indivíduos a serem evoluídos durante o ciclo evolutivo nos algoritmos genéticos está apresentada na figura 5.3.

FIGURA 5.3 - Representação de um Cromossomo em AG's



Os indivíduos são compostos por oito turmas que necessitam da alocação de recursos (disciplinas e professores). Cada turma está dividida em 5 dias semanais e 3 horários diários de aula. A população evoluida no ciclo Genético é composta por indivíduos que representam grades completas para os oito períodos do curso.

Para que a população inicial apresentasse uma boa qualidade em relação às restrições impostas pelo problema, o processo de inicialização das populações é idêntico ao processo de inicialização descrito pelo algoritmo cooperativo. Assim, o processo que gera a população inicial respeita, as restrições de aulas por disciplina e carga horária máxima de aulas por professor para instanciação dos indivíduos. A função de avaliação do *fitness* e os operadores genéticos de seleção, mutação e elitismo também seguem o mesmo processo descrito pela abordagem anterior.

O operador genético de *crossover* também segue o método PMX, porém os pontos de corte sorteados são as turmas para que indivíduos inválidos não fossem produzidos. Assim, a seção de cruzamento precisa ser mapeada de forma que turmas completas sejam cruzadas. Como as disciplinas sempre são diferentes entre as turmas, somente disciplinas pertinentes a cada turma serão cruzadas.

Este capítulo limitou-se a apresentar a maneira com a qual Algoritmos Cooperativos e Algoritmos Genéticos foram utilizados para implementar um software para a geração de Grades Horárias (estudo de caso). Na próxima seção são apresentados os resultados obtidos através de experimentos realizados em ambas as aplicações.

## 6 EXPERIMENTAÇÃO E RESULTADOS

### 6.1 Introdução

Esta fase do desenvolvimento do trabalho trata da validação da implementação computacional do sistema. Sendo a proposta deste trabalho a implementação de um sistema Coevolutivo para produção automática de Grades Horárias, é necessário neste momento, verificar os resultados obtidos com a ferramenta desenvolvida.

Segundo ALENCAR (2001), para conduzir experimentos a uma aplicação computacional que está sendo submetida à verificação, deve-se verificar se a aplicação faz o que era pretendido, ou seja, funciona adequadamente e porque a aplicação funciona. A primeira questão envolve a expressão “funcionar adequadamente” que diz respeito à aplicabilidade das soluções encontradas pelo *software* e já para o segundo aspecto deve-se, de posse dos resultados gerados, elaborar conclusões sobre os mesmos, e sobre o processo que envolve a produção destes com vistas à validade e eficiência do método adotado para a solução do problema.

O *hardware* utilizado para os testes pode ser caracterizado como: Intel Pentium IV CPU 1.80 GHz de frequência de operação; 256 MB RAM; HD com capacidade de 40GB. A plataforma de *software* foi a seguinte: Sistema Operacional Windows XP profissional; Pacote Microsoft 98; Microsoft® Developer Studio Visual C++ 5.0.

#### 6.1.1 METODOLOGIA

Os experimentos foram projetados de forma que a validade e qualidade das soluções apresentadas pelo método implementado, pudessem ser avaliadas. Para viabilizar a verificação da eficácia do método frente a diferentes situações para o estudo de caso, foi levada em conta uma restrição crítica que influencia diretamente na factibilidade e qualidade dos indivíduos, ou seja, a preferência dos horários por parte dos professores.

Assim, diferentes amostras de tabelas de preferência foram elaboradas. Porém para que estas amostras fossem geradas, uma análise mais profunda da influência da disponibilidade de horários na elaboração e validade das grades horárias foi efetuada. Nesta análise pode-se perceber que para a obtenção de uma escala de horários factíveis, dois aspectos devem ser observados:

1. O limite mínimo de horários disponíveis para elaboração das escalas;
2. Uniformidade na distribuição de horários disponíveis dentro da grade horária;

O aspecto 1 é o resultado da observação do número de disciplinas ministradas por cada professor. Cada disciplina estabelece uma carga horária semanal que deve ser cumprida. Portanto cada professor deve disponibilizar no mínimo este número de horas para que as tabelas horárias possam ser viabilizadas.

O aspecto 2 estabelece que se horários forem disponibilizados uniformemente de forma a preencher diferentes posições da escala, respeitando um número mínimo de disponibilidades, o processo de busca torna-se mais fácil. O processo pode ser prejudicado quando as disponibilidades são estabelecidas de maneira não uniforme tornando às vezes a busca da solução impossível.

O que pode ocorrer quando vários professores preferem o mesmo horário da tabela é a falta de professores para preencher a grade horária semanal das turmas. Desta forma a produção de grades horárias factíveis se torna impossível uma vez que faltam recursos para a elaboração das mesmas.

É importante ressaltar a configuração definida na modelagem da restrição *Professor\_Horário*, que preferências de horários foram expressas em termos numéricos que variam de 1 a 5. Estes números são proporcionais ao grau de indisponibilidade de determinado professor, assim, quanto maior o número, menor é a preferência do professor por determinado horário.

Para representar a disponibilidade de horário de cada um dos professores, foram elaboradas arbitrariamente dez amostras, dispostas em ordem de complexidade de acordo com os dois aspectos acima. É importante esclarecer que para períodos em que os referidos professores não ministram quaisquer disciplinas todos os horários foram definidos com valor 5, ou seja, se um professor A só ministrar aulas no período noturno,



então o período vespertino é preenchido totalmente com valores 5 impedindo que o mesmo seja alocado em outro período que não o que realmente é destinado as suas aulas. A representação das amostras é apresentada no anexo A. Assim, as dez amostras podem ser descritas da seguinte forma:

**Amostra 0 :** Apresenta para todos os horários de todos os professores o valor 1, ou seja, disponibilidade total menos para os períodos indisponíveis onde o valor foi definido com 5. Esta situação ocorre, no mundo real, quando os professores tem dedicação integral para o curso, e caracteriza a situação menos complexa para montagem de uma escala de horários. Poder-se-ia em outras palavras dizer que esta situação é a situação ideal para a composição de uma escala.

**Amostra 1:** Respeita o aspecto 1 e 2, disponibilizando além do limite mínimo de horários livres mais oito horários por professor. Esta amostra é composta pelos valores 1 e 2 e 5 para horários indisponíveis.

**Amostra 2:** Respeita igualmente os aspectos 1 e 2, e disponibiliza adicionalmente três horários por professor.

**Amostra 3:** Respeita os aspectos 1 e 2, com disponibilidades adicionais apenas para professores que lecionam mais de uma disciplina.

**Amostra 4:** Respeita os aspectos 1 e 2. É composta somente pelos valores 1, 3 e 5 para horários indisponíveis, procurando obter-se complexidade na busca da solução através da incidência de valores 3 ao invés de 2, o que penaliza mais os indivíduos que tem sua aptidão atribuída pela função de *fitness*.

**Amostra 5:** Respeita os aspectos 1 e 2 e é composta somente pelos valores 1, 4 e 5 para horários indisponíveis . procurou-se estabelecer uma complexidade ainda maior que na amostra anterior pela incidência do valor 4 ao invés de 3 (tomado em conta no calculo do *fitness*).

**Amostra 6:** Respeita os aspectos 1 e 2 porém é composta por valores de preferência 1, 3, 4 e 5 para horários indisponíveis.

**Amostra 7:** Não respeita o aspecto 1, ficando aquém do limite mínimo de disponibilidades para a produção de grades horárias.

**Amostra 8:** Desrespeita o aspecto 2 que está ligado a uniformidade de distribuição de disponibilidades.

**Amostra 9:** Respeita os aspectos 1 e 2 e porém é composta somente por valores 1 e 5 . É válido lembrar que o valor cinco penaliza o indivíduo em 5 vezes o total de restrições violadas até então. Este valor caracteriza insatisfação plena do professor em relação ao horário, caracterizando uma situação proibitiva.

Para cada tabela e gráfico apresentado, os resultados foram calculados com base na média de 10 execuções do sistema. O critério de parada, adotado para cada execução do algoritmo foi à obtenção do *fitness* colaborativo máximo, ou seja, igual a 1. As dez amostras acima descritas, bem como o relacionamento entre disciplinas, professores e turmas é apresentado no anexo A.

## **6.2 Ajuste de parâmetros.**

Quando parâmetros genéticos estão adequadamente ajustados, os resultados apresentados pelo ciclo evolutivo demonstram qualidade há um tempo e custo computacional, relativamente baixos. Para que ocorra este ajuste devem ser executados testes, com diferentes configurações destes parâmetros, procurando estabelecer e analisar as condições ideais para atingir estes resultados.

Entre as dez amostras criadas, fora escolhida para execução destes testes, a amostra 2. Esta amostra promove um nível médio de complexidade de busca, permitindo assim analisar o comportamento do método Cooperativo diante da variação de taxas dos parâmetros genéticos.

### **6.2.1 TAMANHO DA POPULAÇÃO**

O tamanho da população que irá ser tratada em um ciclo evolutivo influencia diretamente no desempenho do algoritmo, pois está ligada ao espaço de busca disponibilizado ao processo.

Assim, quanto maior o número de indivíduos disponíveis ao ciclo evolutivo, maior é o espaço de busca a ser tratado, implicando em um maior custo computacional e mais tempo de processamento. Caso o número de indivíduos da população seja pequeno, a solução pode demorar a ser encontrada ou até mesmo nunca ser atingida, pois fornece pouca cobertura no espaço de busca.

Para verificar o comportamento do sistema ante a variação do número de indivíduos existentes nas populações fixou-se, *a priori*, o valor dos seguintes parâmetros genéticos: *crossover* 50, mutação de 30 e elitismo de 30. Em 10 execuções para cada variação de número de indivíduos obteve-se a tabela 6.1.

TABELA 6.1- Número de gerações obtido com variação do tamanho da população

Tamanho da população	Número de Gerações
300	70
500	62
700	47
900	26
1000	22

Na tabela 6.1, pode-se observar que quanto maior o tamanho da população a ser tratada, menor é o número de gerações necessárias para a obtenção da solução ótima. Isto se deve ao fato de que com uma população maior, a diversidade de indivíduos também aumenta o que leva o processo a convergir mais rapidamente para uma solução ideal. Quando tem-se uma população muito grande, superior ao valor de 1000, o tempo gasto para o ciclo evolutivo gerar a solução ótima aumenta consideravelmente.

Portanto, o tamanho da população será fixo para os próximos experimentos em 1000 indivíduos, por ter apresentado um número relativamente pequeno de gerações necessárias e constituir um espaço amostral suficiente para obtenção da solução ótima.

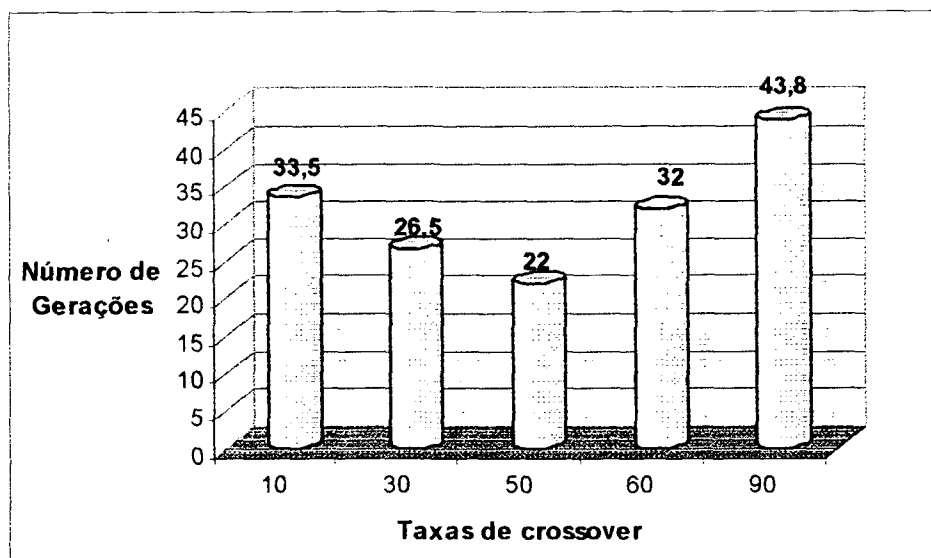
#### 6.2.2 TAXA DE CROSSOVER

O operador de *crossover*, como já mencionado anteriormente, permite a troca de material genético e a conseqüentemente produção de novos indivíduos, permitindo

portanto o aumento da diversidade na população. Quanto maior é a probabilidade de incidência de *crossover* sobre os indivíduos de uma população, novas estruturas serão inseridas mais rapidamente na população.

Porém, quanto este valor é muito elevado, pode ocorrer um efeito indesejado uma vez que a substituição constante de novos indivíduos na maioria da população perde-se na variedade genética e quando a taxa é muito reduzida o processo evolutivo pode tornar-se lento para produção de soluções ótimas, então através do experimento objetiva-se encontrar um equilíbrio. Tem-se na figura 6.1, os resultados dos experimentos.

FIGURA 6.1 - Número de Gerações em função da variação da taxa de *crossover*.



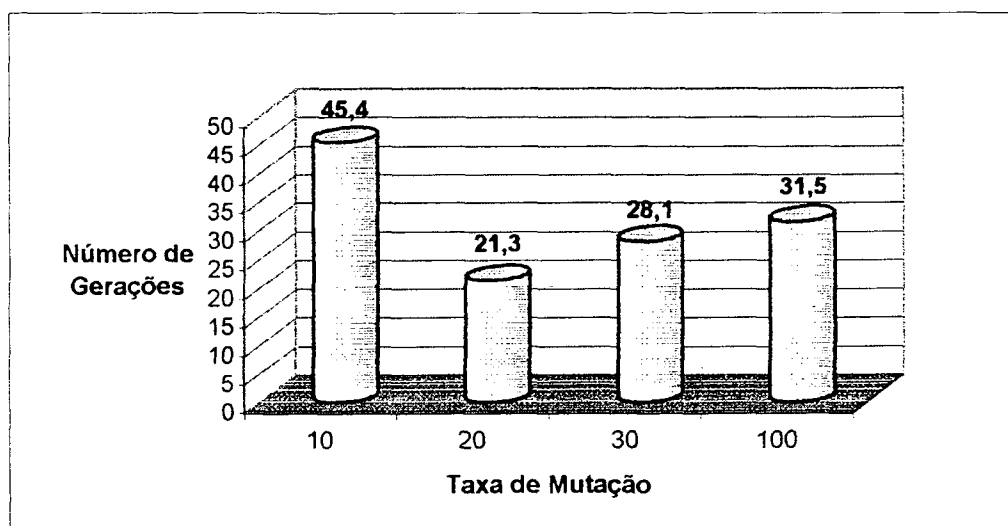
Como afirmado anteriormente, após a verificação dos experimentos apresentados na figura 6.1 observa-se que quando a taxa de *crossover* é muito alta ou muito baixa, o algoritmo precisou de mais gerações para que o processo evolutivo encontrasse a solução ótima.

No caso da taxa intermediária de 50, o processo apresentou resultados bons, garantindo soluções ótimas. Em consequência dos resultados observados nos experimentos desta seção, optou-se por fixar para os próximos experimentos a probabilidade de incidência do operador genético de *crossover* à taxa de 50%.

### 6.2.3 TAXA DE MUTAÇÃO

O processo de mutação, conforme citado anteriormente, promove alterações genéticas nos indivíduos, prevenindo assim a estagnação do *fitness* da população. Quando a taxa de mutação é muito alta o processo de busca pode tornar-se essencialmente aleatório o que dificulta e retarda a produção da solução ótima. Os experimentos realizados no Sistema Cooperativo implementado estão representados na figura 6.2.

FIGURA 6.2 - Número de gerações em função da variação da taxa de mutação.



Na figura 6.2 pode-se confirmar a afirmativa que prediz que taxas altas aumentam o número de gerações necessárias para a obtenção da solução ótima, pois a busca é essencialmente aleatória. No caso de taxas pequenas de mutação (10), a diversidade genética apresentada pelos indivíduos é prejudicada, permitindo que a ocorrência da estagnação do *fitness*.

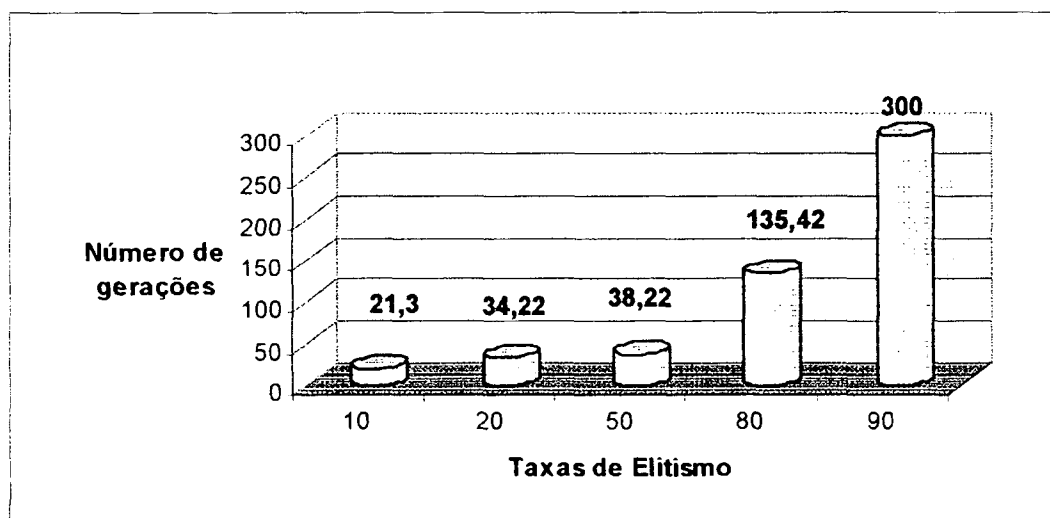
O ideal, portanto, é estabelecer um equilíbrio entre estes dois extremos, expresso através dos resultados apresentados na figura com o valor de 20. Adota-se portanto para os demais testes o valor da taxa de mutação de 20%.

### 6.2.4 TAXA ELITISMO

Nos algoritmos evolutivos existe o processo conhecido por seleção salvacionista ou elitismo que mantém indivíduos com alto *fitness* de uma geração atual em gerações

posteriores. Os indivíduos selecionados não passam nas gerações posteriores pelo processo de mutação, mas podem gerar novos filhos e serem substituídos por indivíduos de *fitness* melhor. Para a variação desta taxa no algoritmo coevolutivo implementado, obteve-se comportamento descrito pelo gráfico 6.3.

FIGURA 6.3 - Número de Gerações em função da variação da taxa de elitismo.



É possível se observar na figura 6.3 que quanto maior é o número de indivíduos mantidos pelo elitismo, maior é o número de gerações necessárias à obtenção de uma solução ótima. Isto ocorre, porque em várias gerações os mesmos indivíduos aparecerem na população, prejudicando o ciclo evolutivo em termos de diversidade. Nos casos onde a taxa elitista é muito alta acontece o fenômeno conhecido na literatura como convergência prematura, abordado anteriormente.

Portanto, para os experimentos da próxima seção, a faixa de elitismo dos indivíduos nas populações compreenderá 10% do total da população, pois promoveu bons resultados no desempenho do algoritmo.

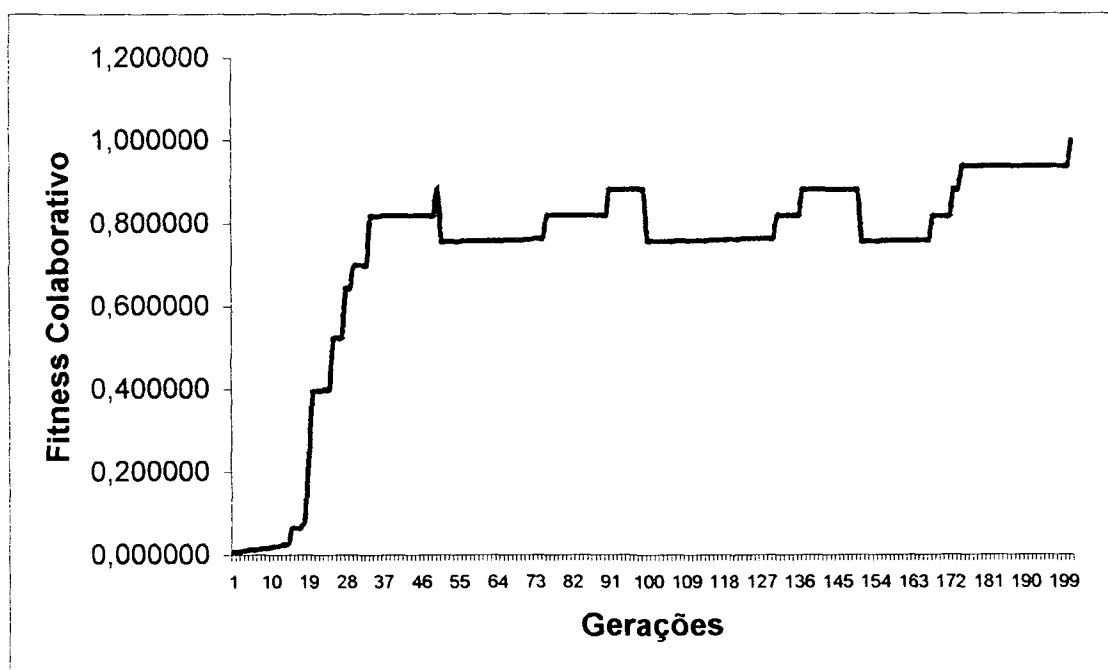
É importante esclarecer que a taxa de *crossover*, mutação, elitismo e todas as demais, pode ser considerada proporcional às taxas utilizadas na grande maioria dos AG's. Os valores de 20, 50 e 10 foram consequência da formalização dos cálculos realizados dentro do sistema que em nada prejudicaram o ciclo evolutivo implementado.

### 6.2.5 EVOLUÇÃO DO *FITNESS*

É interessante observar o comportamento que as populações apresentam durante o processo evolutivo, com vistas ao seu valor de *fitness*. Relembra-se aqui que o valor máximo para o *fitness* colaborativo é igual a 1.

Para os testes desta seção, a amostra 6 foi escolhida, pois, devido a sua complexidade, esta amostra requer um número maior de inicializações. Assim será possível observar além da evolução do *fitness*, as variações na curva de *fitness* quando ocorrer alguma reinicialização de populações estagnadas. A evolução ou curva de *fitness* apresentada em uma execução do algoritmo Cooperativo implementado, está apresentada na figura 6.4.

FIGURA 6.4 – Evolução de *fitness* para a amostra 6.



Pode-se observar através da figura 6.4 que, à medida que o ciclo evolutivo foi realizado, o *fitness* colaborativo das populações foi aumentando gradativamente. Disto conclui-se que à medida que operadores genéticos vão sendo aplicados nos indivíduos das populações todas as restrições, consideradas no cálculo do *fitness*, vão sendo satisfeitas.

Assim, a solução ótima sempre gerará, como no caso do gráfico figura 6.4, uma solução que traz como preferência para todos os professores os horários expressos pelo

valor de 1, sendo que outras restrições como carga horária de disciplinas e professores também são respeitadas. A não existência de restrições nos representantes das populações, atingiu na geração 201 o *fitness* colaborativo ótimo, ou seja, valor 1,000000 sendo a condição que encerra o ciclo evolutivo.

Além disso, pela figura 6.4 é possível perceber que a reinicialização das populações estagnadas contribui muito para o sucesso do ciclo evolutivo. Logo que uma população é reinicializada, a cada 50 gerações como no caso do algoritmo implementado, o *fitness* colaborativo decresce um pouco, porém, após algumas aplicações de operadores genéticos, que promovem o melhoramento dos indivíduos através das gerações, o *fitness* colaborativo evolui. O processo evolutivo foi encerrado quando o valor máximo do *fitness* colaborativo (1) foi atingido na população 201.

Pode-se observar que na geração 40 o *fitness* colaborativo era de 0.82 sendo que na geração 200 o *fitness* finalmente chega ao valor máximo. A observação valor obtido na geração 40 a primeira vista pode provocar certas indagações quanto ao ganho real da qualidade obtida ao final das duzentas gerações. Esse ganho consiste na satisfação plena da restrição de preferências de horários, ou seja, na quadragésima geração nem todos os professores estavam alocados nos horários desejados, por isso o *fitness* não tinha atingido o valor máximo. Esse valor foi obtido somente na geração 200 onde a *timetable* satisfaz plenamente todas as restrições impostas pelo problema.

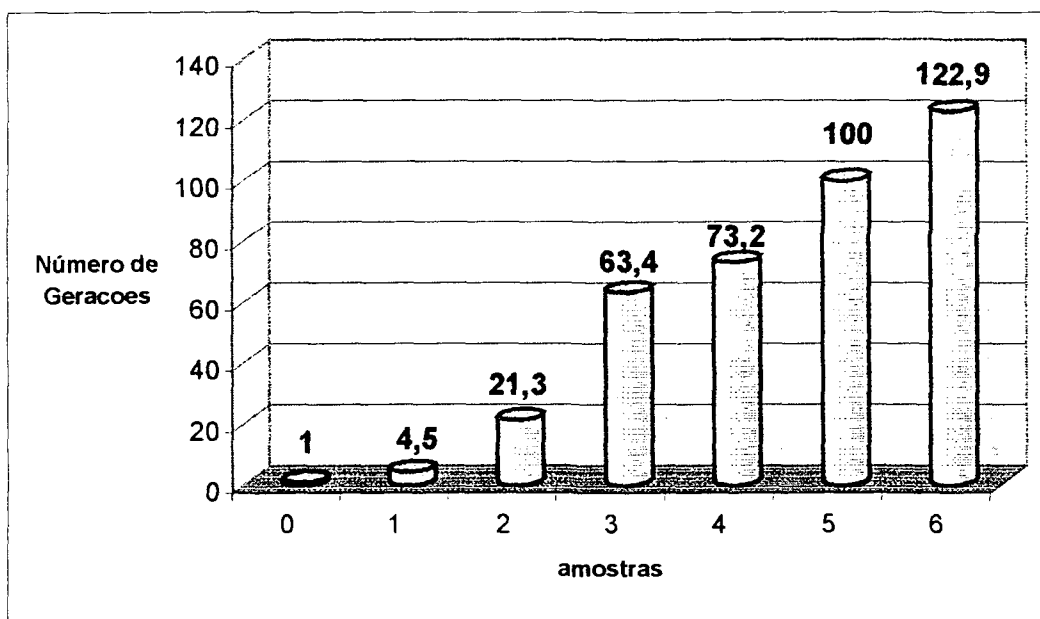
#### 6.2.6 ALGORITMO COOPERATIVO X AMOSTRAS DE DISPONIBILIDADE DE HORARIOS

A análise a ser realizada nesta seção procura investigar comportamento do sistema Cooperativo quando as restrições a que o problema está sujeito se tornam mais “pesadas”, tornando assim o processo de produção das grades horárias mais complexo e custoso.

Para a viabilização destes testes, fez-se uso das dez amostras descritas anteriormente. Os parâmetros genéticos, estabelecidos anteriormente, permanecem fixos da seguinte forma: tamanho da população de 1000 indivíduos, taxa *crossover* foi estabelecida em 50%, taxa de mutação em 20% e taxa de elitismo de 10%. Em dez execuções do algoritmo, se obteve os resultados apresentados na figura 6.5. Os resultados obtidos nas amostras de número 7, 8 e 9 foram omitidos por motivos a serem descritos posteriormente nesta seção.



FIGURA 6.5 - Número de Gerações obtido na Evolução Cooperativa frente a diferentes níveis de complexidade na busca.



É importante esclarecer que o número de gerações apresentado nas figuras é as vezes fracionário pois foi calculado com base na média de 10 execuções do sistema. Esta média resulta, na grande maioria das vezes em valores não inteiros, ocasionando em número de gerações apresentados nas figuras.

Na figura 6.5, a amostra 0 apresentou o menor número de gerações necessário. Isto ocorre porque, nesta amostra, a disponibilidade dos professores é total, fazendo com que a solução ótima seja encontrada mais rapidamente.

Nas amostras onde a complexidade imposta ao processo de busca era pequena, amostras de 1 a 3, a solução ótima foi rapidamente produzida. Deste resultado, pode-se observar que, quando as disponibilidades são distribuídas de maneira uniforme e em quantidade superior ao limite mínimo exigido, a produção de soluções acontece de maneira mais rápida. À medida que esse é limite atingido, o processo necessita de mais gerações, porém, pelo método implementado, a solução ótima continua sendo facilmente atingida.

Nos testes 4, 5 e 6 o nível de complexidade se eleva através de menor disponibilidade dos professores. Nestas amostras o valor de preferência 1 foi mantido

apenas para respeitar o limite mínimo de disponibilidades e todo o restante das preferências foram definidas com valores 3, 4 e 5. Portanto o número de gerações teve de ser maior para que uma solução ótima fosse atingida. Contudo como, o limite mínimo de disponibilidades foi respeitado, mesmo em número maior de gerações, soluções ótimas foram obtidas.

Quando o Algoritmo Cooperativo foi aplicado na amostra 7 e 8, que não respeitam o limite mínimo de horários disponíveis e uniformidade respectivamente, não conseguiu obter uma solução ótima, pois nestes casos faltaram recursos. O *fitness* colaborativo máximo atingido para estas amostras foi 0,753423 e 0,635249 respectivamente. Conforme citado anteriormente, quando estes dois aspectos não são respeitados a produção de soluções ótimas é inviabilizada pois faltam recursos para completude das mesmas. Além disso, o Algoritmo Cooperativo não tem autonomia para disponibilizar os horários definidos por parte dos professores.

Finalmente, quando a amostra 9 foi aplicada ao algoritmo, nem mesmo o processo de inicialização das populações foi concluído. Isto ocorreu porque estas amostras contêm grande quantidade de valores 5 para horários. Este valor caracteriza que o professor não disponibilizou o referido horário para lecionar aulas. No processo de inicialização das populações não é permitido que horários indisponíveis sejam escalonados nas grades criadas, assim, a geração das populações não pode ser concluída. As soluções fornecidas pelo algoritmo, para as amostras de 0 a 6, são apresentadas no anexo B.

Outro aspecto interessante a ser observado é a necessidade da verificação de colaboração entre as populações no processo evolutivo, que consiste em um importante objetivo dos Algoritmos Cooperativos. Caso alguma população esteja estagnada, a mesma é eliminada do ambiente e novos indivíduos são inseridos no meio através do processo de inicialização da população. Na tabela 6.2, este número será expresso para as amostras 3, 4, 5 e 6. Para as amostras de número 0, 1 e 2 o valor foi omitido pois o número de populações estagnadas foi zero pela baixa complexidade estabelecida no processo de busca, e para as amostras 7, 8 e 9, de igual forma pois uma solução ótima não foi obtida.

TABELA 6.2 - Populações estagnadas X Amostras de disponibilidade horária.

Amostra	Nº Populações Estagnadas
3	1
4	1
5	1
6	2

O número máximo de vezes que a colaboração das populações no processo evolutivo é avaliada é de 10 vezes, sendo executado a cada 50 gerações. Baseando-se nas características das dez amostras, pode-se observar que à medida que o fator disponibilidade de horários e uniformidade de distribuição dos mesmos decresce o processo de busca torna-se mais complexo um número maior de populações permanecem com o *fitness* de seus indivíduos estagnado. Assim o processo de inicialização ocorre um número de vezes proporcional ao nível de complexidade do processo de busca.

### 6.3 Comparação com Algoritmos Genéticos

Para a viabilização dos comparativos entre Algoritmos Genéticos e Cooperativos os parâmetros evolutivos como taxa de *crossover*, taxa de mutação e faixa de elitismo foram mantidos.

Contudo, para que nenhum dos dois métodos fosse prejudicado quanto ao número de indivíduos a serem tratados no ciclo evolutivo, atribuiu-se ao tamanho da população na abordagem Genética o valor de 8000. Este número foi concebido uma vez que no Algoritmo Cooperativo são evoluídas 8 populações de 1000 indivíduos cada.

#### 6.3.1 ALGORITMO GENÉTICO X AMOSTRAS DE DISPONIBILIDADE DE HORARIOS

As dez amostras de disponibilidade de horários foram aplicadas no algoritmo Genético, e os resultados obtidos estão expressos na tabela 6.3. Nesta tabela o desempenho do algoritmo Genético está expresso em termos de *fitness* máximo atingido.

Adotou-se esta representação, pois o valor ótimo de *fitness* igual a 1 não foi atingido, salvo os casos das amostras 0 e 1. Neste uma solução ótima foi obtida na geração

14 e 72 respectivamente, devido à baixa complexidade imposta, através da amostra, ao processo de busca.

TABELA 6.3- *Fitness* AG X Amostras de disponibilidade horária.

Amostra	Número da última Geração	<i>Fitness</i> máximo obtido
0	14	1,000000
1	72	1,000000
2	500	0,250000
3	500	0,011628
4	500	0,007692
5	500	0,007752
6	500	0,006452
7	500	0,003571
8	500	0,004032
9	0	-

É interessante observar na tabela 6.3 que, de uma forma geral, quanto mais se aumentou a complexidade para as tabelas de preferência de horário, menor foi o *fitness* obtido pelo processo evolutivo. Isso ocorreu em consequência das restrições violadas pelos indivíduos, penalizando em muito o valor final do *fitness*. Como no algoritmo genético, apesar de usar um processo de *crossover* adequado (PMX) para o problema, indivíduos que possuem um *fitness* baixo podem continuar trocando material genético com os indivíduos melhores. Esta troca faz com que a qualidade genética da população não evolua.

O que se observa no desempenho do Algoritmo Genético, a partir dos experimentos, é que a partir de um número determinado de gerações o *fitness* da população permanece estagnado. Como na abordagem de algoritmos genéticos, não há nenhum processo reverta esta situação, a exemplo da verificação da colaboração cooperativa, os indivíduos demoram a evoluir. Mesmo quando a taxa de parâmetros genéticos como mutação e *crossover*, tamanho da população é grande uma solução ótima não pode obtida por falta de recursos auxiliares.

Portanto, de acordo com os resultados verifica-se que algoritmos genéticos clássicos não são métodos eficientes para a resolução de problemas complexos como *timetabling*. A mesma situação não ocorre para algoritmos cooperativos, uma vez que estes suprem as deficiências apresentadas por Algoritmos Genéticos, podendo, portanto resolver problemas mais complexos e com um maior número de restrições.

#### **6.4 Contribuições obtidas através do método Coevolutivo**

Com vistas aos resultados apresentados na seção de modelagem do problema, pode-se verificar que o método Coevolutivo melhora muito a representação das soluções candidatas para problemas complexos sujeitos a restrições. Isto porque o problema original pode ser subdividido em partes menores e mais fáceis de serem solucionadas. Quando todos os subproblemas forem resolvidos, tem-se a solução para o problema original.

Como exemplo disto, retoma-se a representação utilizada por NEWALL (2000) para *timetabling* para exames em universidades, abordado no capítulo 4 deste trabalho. Este problema compreende muitas restrições que variam de universidade para universidade. Os recursos envolvidos na elaboração das soluções podem ser descritos como: períodos ou horários de aplicação de exames, salas de aula e exames propriamente ditos. De acordo com os critérios dados por cada instituição, pode-se definir uma *timetable* como factível ou não.

Transportando o problema para a arquitetura cooperativa poderia se definir uma espécie como sendo cada um dos períodos ( $p_i$ ) de exames ( $e_j$ ). Assim, os indivíduos seriam representados por tabelas instanciadas com salas disponíveis e respectivos exames relacionados. Essa representação está apresentada na figura 6.6.

FIGURA 6.6 - Representação Cooperativa de um indivíduo para *timetable* para exames.

Período 1	
Sala 1	$e_1, e_2, e_4$
Sala 2	$e_3, e_7$
Sala 3	$e_{11}, e_{36}$
Sala 4	$e_{90}, e_{27}$

O número de espécies a serem inseridas no sistema é equivalente ao número de períodos de exames envolvidos no problema em questão. Todas as populações ou espécies durante o processo evolutivo, vão cooperar para formar uma solução completa que satisfaça todas as restrições do problema. Além disso, esta representação garante que durante a aplicação de operadores genéticos, somente características inerentes ao próprio período sejam trocadas pelas restrições de isolamento impostas pela arquitetura cooperativa, evitando assim a infactibilidade nos filhos produzidos.

Outro aspecto interessante da arquitetura modelada pela Evolução Cooperativa é que as populações pertinentes ao sistema são forçadas a cooperar durante o ciclo evolutivo. Esta cooperação é imposta pela função de *fitness* e verificação de estagnação do mesmo. Como as populações estagnadas são reinicializadas, novas e maiores chances da obtenção de uma solução adequada às restrições do problema são oferecidas ao ciclo evolutivo.

Os processos de cruzamento e mutação implementados no algoritmo desenvolvido só vieram a contribuir para o objetivo inicial do método Cooperativo no tratamento de *timetable*. Além disso, o método permitiu, com sucesso, a modelagem e resolução do problema de elaboração de Grades Horárias, obtendo em várias situações respostas satisfatórias. Prova disto são os resultados apresentados pelo algoritmo Cooperativo implementado, atribuindo assim a validade do método implementado na resolução do problema analisado.

## 7 CONSIDERAÇÕES FINAIS E PERSPECTIVAS FUTURAS

A computação evolutiva descreve métodos e técnicas que têm sido estudadas e aperfeiçoadas ao longo de décadas por vários pesquisadores. Esta abordagem provê uma série de recursos e, em consequência disto os métodos evolutivos são reconhecidos como ferramentas poderosas para a resolução de vários tipos de problemas.

Os Algoritmos Genéticos consistem de uma técnica evolutiva utilizada geralmente para solucionar problemas de otimização numérica. Algumas técnicas alternativas em AG têm sido propostas para trabalhar com espaços de busca sujeito a restrições, dentre estes métodos destacam-se: Heurísticas Dirigidas, Grafos Coloridos, Programação Lógica e Funções de Penalidade.

Neste trabalho foi apresentado um estudo da aplicação de Algoritmos Coevolutivos para *Timetabling* (problema de otimização numérica sujeito a restrições). Este método é na verdade uma extensão de Algoritmos Genéticos, porém na abordagem Coevolutiva, os indivíduos são divididos em espécies.

Os resultados obtidos na aplicação de algoritmos Coevolutivos e Algoritmos Genéticos, apresentados no capítulo de experimentos, conclui-se que AG's na abordagem clássica encontram dificuldades no tratamento de problemas de otimização com muitas restrições como *timetabling*, ao passo que os Algoritmos Coevolutivos, forneceram resultados satisfatórios. Deduz-se que este resultado foi obtido em consequência de alguns recursos adicionais importantes providos pelo método cooperativo.

O primeiro recurso diz respeito à divisão de espécies. Como no ciclo evolutivo, as espécies são processadas de maneira paralela, e indivíduos que tiverem um valor de *fitness* baixo não prejudicarão indivíduos que pertençam a uma outra espécie durante o ciclo evolutivo.

Neste mesmo contexto, é importante destacar que apesar da modularização realizada no ecossistema pelo método Coevolutivo, as espécies precisam evoluir de maneira cooperativa. Isto quer dizer que a divisão do ecossistema em espécies não implica

na independência entre as mesmas pois a avaliação dos indivíduos é feita em conjunto com os representantes das demais populações. Assim todos os indivíduos devem cooperar para a obtenção do objetivo.

Outro recurso que melhorou em muito o desempenho do ciclo no método Coevolutivo foi a verificação da estagnação do *fitness* das espécies. A reinicialização das populações improdutivas fez com que o *fitness* da população, antes estagnado, revertisse esta situação através da inserção de novos indivíduos no ecossistema. Assim, uma nova chance à evolução do *fitness* é proporcionada, permitindo mesmo em casos onde o espaço de busca é restrito, uma solução ótima fosse obtida.

Considerando como desvantagem do sistema implementado a especificidade do aplicativo, para trabalhos futuros sugere-se a generalização do método implementado para que a aplicação do mesmo a outras variantes de *timetabling* como escalas de funcionários ou remanejamento de máquinas seja possível.

Outro aspecto a ser considerado em estudos futuros é a “queda” do *fitness* colaborativo quando da reinicialização das populações improdutivas, visto no capítulo de experimentos. Essa “queda” faz com que o ciclo demore mais a encontrar uma solução ótima pois ocasiona um certo retardo no processo até que o *fitness* da espécie reinicializada atinja o valor anterior, permitindo assim evolução do *fitness* colaborativo.

Desta forma, em trabalhos futuros, pode-se desenvolver um processo de inicialização específico para populações estagnadas. Neste processo, deve-se evitar, de alguma forma, a queda imediata do *fitness*, e, ao mesmo, tempo fazer com que o quadro a estagnação possa ser revertida.

Seria interessante ainda realizar a comparação dos resultados obtidos para quaisquer problemas de otimização através do método cooperativo, com outros métodos não evolutivos. Poderia se analisar métodos como a Programação Linear, através de ferramentas como LINGO, ou ainda estratégias de busca convencional como A\*.



## REFERÊNCIAS BIBLIOGRÁFICAS

ALENCAR, W. S. **HORUS, Sistema de elaboração automática de aulas combinando programação lógica com restrições hierárquicas.** Dissertação de Mestrado em Engenharia Elétrica e de Computação. Universidade Federal de Goiás – Goiânia, 2001.

BACK, T.; FOGEL, D.B.; MICHALEWICZ, Z. **Handbook of Evolucionary Computation.** Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

BARBOSA, H. **An Adaptive Penalty Scheme In Genetic Algorithms For Constrained Optimization Problems.** In Proceedings of the Genetic and Evolutionary Computation Conference. Morgan Kaufmam Publishers. New York. 2002

BURKE, E.K ; NEWALL, J.P; WEARE, R.F. **A memetic algorithm for university exam Timetabling.** In Edmund Burke and Peter Ross, editors, The practice and theory of automated *Timetabling*: Selected papers from de 1<sup>st</sup> international conference, lecture notes in computer science 1153, pages 241-250. Springer- Verlag, Berlim, 1996.

CARVALHO, A. C. P. L. F; LACERDA, E.G.M. **Introdução aos Algoritmos Genéticos.** Departamento de Informática UFPE e ICMC/USP, 1999.

CONCILIO, R. **Contribuições à solução de problemas de Escalonamento pela Aplicação Conjunta de Computação Evolutiva e Otimização de Restrições.** Dissertação apresentada ao Departamento de Engenharia da Computação e Automação Industrial (DCA) Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas UNICAMP-SP, 2000.

COSTA, Eduardo O.; BRUNA Marlonn D. **Resolução de Timetabling utilizando algoritmos Genéticos e Evolução Cooperativa.** Trabalho de Graduação em Inteligência

Artificial apresentado ao curso de Ciências de Computação, Departamento de Informática da Universidade Federal do Paraná-UFPR. Curitiba. 2002

DAWNKIS, R. **The Selfish Gene**. Oxford University Press, 1976.

GOLDBERG, D. E. **Genetic Algorithms in Search, Optimization, and Machine Learning**. Addison-Wesley, 1989.

HOLLAND, J.H. **Adaptation in natural and artificial systems**. Ann Arbor: The University Of Michigan Press, 1975, in Goldberg Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.

HOLLAND, J.H. **Genetic Algorithms and adaptation**, 1984, in Goldberg Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.

KANG, L; WHITE, G.M. **A logic approach to the resolution of constraints in Timetabling**. European Journal of Operational Research. 61, 306-317. 1992

MAIA, A. C. P. ; MICHELAN, R., **Algoritmos Genéticos**. Agosto de 2001. <http://www.din.uem.br/~ia/geneticos/index.htm>

MITCHELL, M. **An Introduction to Genetic Algorithms**. The MIT Press, 1998.

MOSCATO P. ; NORMAM M.G. **A memetic approach for the travelling salesman problem- implementation of computational ecology for combinatoria optimisation on messge-passing systems**. IOS Press (Amsterdam), 1991.

NARDIN, L. D. **Estudo da Aplicação de Algoritmos Genéticos Para Resolução do Problema de Geração de Horários**. Trabalho de Graduação em Inteligência Artificial apresentado ao curso de Informática na Universidade Estadual do Oeste do Paraná-UNIOESTE Novembro, 1999 .

NEWALL, J.P. **Hybrid Methods for Automated *Timetabling***. PhD Thesis, University of Nottingham, Department of Computer Science. UK, 2000.

OLIVEIRA, A.C. **Algoritmos Evolutivos para problemas de otimização numérica com variáveis reais**. Monografia apresentada ao exame de qualificação do Curso de Computação Aplicada – CAP São José dos Campos. SP. 2001

POTTER M.A., JONG.K.A. **Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents**. In *Evolutionary Computation*. MIT Press, 2000.

ROSS P., FANG H. , CORNE D. **Genetic Algorithms for *Timetabling e scheduling***. Departamento de IA Universidade de Edinburgh, 80 South Brige, Edinburgh EH 11HN, Scotland. 1999.

ROSS, P.; FANG, H. ; CORNE, D.. **Departamento de IA Universidade de Edinburgh, 80 South Brige, Edinburgh EH 11HN, Scotland**. Fast Pratical Evolutionary *Timetabling*, 2000.

SCHAERF, A. **A Survey of automated *timetabling***. Computer Science/Departament of Software and Technology (CSR-967) 1995.

YEPES, Y. **Uma Incursão Aos Algoritmos Genéticos. Projeto Ísis**. Novembro de 2001.  
<http://www.geocities.com/igorvapes/>

ZIZTLER, E. **Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications**. Dissertation submitted to the Swiss Federal Institute of Technolgy Zurich, November 1999.

# **Anexo A**

## Preferência dos Professores com relação aos horários de aulas.

Abaixo estão apresentadas as nove amostras de disponibilidade de horário criadas para a execução dos experimentos realizados neste trabalho. Nas colunas 0 a 30 estão representadas as preferências horárias para cada um dos 34 professores, representados nas linhas. É importante lembrar que as preferências foram expressas em uma escala numérica de insatisfação que vai de 1 a 5, onde o valor 1 representa os horários disponíveis e 5 não disponíveis. As colunas 0 a 14 representam os horários matutinos e as demais colunas os horários noturnos.

### Amostra 0

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2 Hexse	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5 CDIII	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9 Andre	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10 Estll	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16 PO	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
28 Urban	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
31 Opt1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

# Amostra 1

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	2	1	1	1	2	1	1	1	1	1	2
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	2	1	1	1	1	2	1	1	1	1	2
2 Hexse	2	1	1	1	1	1	2	1	2	2	1	1	2	2	1	1	1	1	1	1	2	1	1	1	1	2	1	2	2	2	2
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	2	1	2	1	1	1	2	1	1	1	1	2
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
5 CDIII	1	2	1	1	1	1	2	1	1	2	1	1	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	1	1	1	2	1	1	1	1	1	2	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	2	2	2	2	1	1	1	2	2	1	1	1	2	2	1	1	1	2	2	1	2	1	2	2	2	2	1	1	2	2	2
9 Andre	1	2	2	2	2	2	1	2	2	2	2	2	1	1	2	1	1	1	1	1	2	1	2	1	1	2	2	2	1	2	2
10 EstII	1	1	1	1	1	1	1	1	2	2	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	1	2	1	1	1	1	1	1	1	2	1	2	2	2	1	1	1	2	1	1	1	1	1	1	1	2	1	1	1	2	2
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	2	1	1	2	1	2	1	1	1	1	2	2
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	2	2	1	1	1	2	2	2
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	2	1	1	1	1	1	1	1	2	2	2	2
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	2	1	2	1	1	1	2	1	1	2	2
16 PO	1	1	1	2	2	1	1	1	2	2	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	1	1	1	1	1	1	1	2	1	2	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	1	1	1	1	1	1	1	2	1	1	1	2	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	1	1	1	2	1	1	1	1	2	1	2	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	1	1	1	1	1	1	2	1	1	1	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	1	1	1	1	1	1	1	1	2	2	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	1	1	1	1	2	1	1	1	1	2	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	1	1	1	2	2	1	1	1	2	2	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	2	1	1	1	2	2	1	2	1	2	2
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	2	1	2	2	2	2
28 Urban	1	1	2	1	1	1	1	1	2	2	1	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	1	1	1	1	1	1	2	2	2	1	1	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2
31 Opt1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2

Amostra 2

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	2	1	2	2	1	2	2	1	2	2	1	1	2
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	1	2	1	2	1	1	2	2	2	2	2	2
2 Hexse	2	2	1	1	1	1	2	2	2	2	1	2	2	2	1	1	2	1	1	2	2	1	1	1	1	2	2	2	2	2	2
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	2	1	2	1	2	1	2	1	2	2	2	2
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	1	2	1	2	1	2	2	2	1	2	1	2
5 CDIII	1	2	1	2	1	2	2	1	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	2	1	2	2	1	2	2	1	1	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	2	2	1	1	1	1	1	1	1	1	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	2	2	2	2	1	1	1	2	2	1	2	2	2	2	1	1	2	2	2	1	2	2	2	2	2	2	2	1	2	2	2
9 Andre	1	2	2	2	2	2	1	2	2	2	2	2	1	2	2	1	2	1	2	2	1	2	2	2	2	2	2	2	1	2	2
10 EstII	1	2	2	2	1	2	2	1	2	2	1	2	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	1	2	1	1	1	1	1	2	1	2	1	2	2	2	2	1	1	2	1	2	2	2	1	1	2	1	1	1	1	2	2
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	2	2	2	1	1	2	2	2	2	1	2	2	2	2
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	1	2	2	1	1	2	2	1	2	2	2	2	2
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	1	2	1	2	2	1	1	1	2	2	2	2	2
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	2	2	2	1	2	2	2	1	2	1	2	2	2
16 PO	2	1	2	2	2	2	2	1	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	2	2	1	1	1	2	2	2	1	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	2	2	1	2	1	1	2	2	2	1	1	2	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	2	1	2	1	2	1	1	2	2	1	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	2	1	2	1	2	2	1	2	1	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	2	2	2	2	2	2	1	1	2	1	2	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	1	1	2	2	1	1	2	2	2	2	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	2	2	2	2	1	1	1	1	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	2	1	1	1	2	2	2	1	1	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 EI	2	1	2	2	2	1	1	2	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	1	2	2	1	2	2	2	2	2	1	2	2
27 Eli	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	1	2	2	1	1	1	2	2	2	2	2	2	2
28 Urban	2	1	2	2	1	2	2	1	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	2	1	1	1	2	2	2	2	2	2	2	1	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	2	2	2	1	1	2	2	1	1	1	1	2	2
31 Opt1	2	1	1	1	2	2	1	1	2	1	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	1	1	1	2	2	1	1	1	2	1	1	2	2
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	1	1	1	1	1	1	2	1	2	1	1	2	2

### Amostra 3

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	2	2	1	2	2	1	2	2	1	2	2	2	2	2
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	2	2	2	2	1	2	2	2	2	2	2	2
2 Hexse	2	1	1	1	1	2	2	1	2	2	1	2	2	2	1	2	2	1	1	2	2	1	1	1	1	2	2	2	2	2	2
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	2	1	2	2	2	2	2	1	2	2	2	2
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	2	2	1	2	2	2	2	2	1	2	2	2
5 CDIII	1	2	2	2	1	2	2	2	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	2	1	2	2	2	2	2	2	1	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	2	2	2	1	1	2	1	1	1	2	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	2	2	2	2	2	2	1	2	2	1	2	2	2	2	1	2	2	2	2	1	2	2	2	2	2	2	1	2	2	2	2
9 Andre	1	2	2	2	2	2	2	2	2	2	2	2	1	2	2	2	2	1	2	2	1	2	2	2	2	2	2	2	2	2	2
10 EstII	2	2	2	2	2	2	2	2	2	2	1	2	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	2	2	1	1	2	1	1	2	1	2	1	2	2	2	2	2	1	2	1	2	2	2	1	2	2	1	1	1	1	2	2
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	2	2	2	1	2	2	2	2	2	2	2	2	2	2
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	2	2	2	1	2	2	2	2	2	2	2	2
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	1	2	2	2	2	2	1	2	2	2	2	2	2
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	2	2	1	2	2	2	1	2	2	2	2	2
16 PO	2	1	2	2	2	2	2	1	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	2	2	1	2	2	2	2	2	1	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	2	2	1	2	2	1	2	2	2	1	2	2	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	2	2	2	1	2	2	2	2	2	1	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	2	2	2	2	2	2	1	2	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	2	2	2	2	2	2	2	1	2	2	2	2	1	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	2	2	2	2	1	2	2	2	2	2	1	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	2	2	2	2	1	1	2	2	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	2	1	2	2	2	2	2	1	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	2	2	2	2	2	1	2	2	2	2	2	1	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	2	2	2	1	1	2	2	2	2	2	2	2	2
28 Urban	2	1	2	2	1	2	2	2	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	2	1	1	2	2	2	2	2	2	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	2	2	2	2	2	2	2	2	1	1	2	1	2	2
31 Opt1	2	1	2	2	2	2	2	1	2	1	1	1	1	2	2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	2	2	2	2	1	2	2	1	1	1	2	1	1	2	2
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	1	2	1	2	1	1	2	2	1	2	1	1	2	2	2



**Amostra 4**

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	1	3	3	1	3	3	1	3	3	3	3	3	3
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	3	3	3	3	3	1	3	3	3	3	3	3	3
2 Hexse	3	1	1	1	1	3	3	3	3	3	1	3	3	3	1	3	3	1	1	3	3	1	1	1	1	3	3	3	3	3	3
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3
5 CDIII	1	3	3	3	1	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	3	1	3	3	3	3	3	3	1	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	3	3	3	1	1	3	1	1	1	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	3	3	3	3	3	3	1	3	3	1	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3
9 Andre	1	3	3	3	3	3	3	3	3	3	3	3	1	3	3	3	3	1	3	3	1	3	3	3	3	3	3	3	3	3	3
10 EstII	3	3	3	3	3	3	3	3	3	3	1	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	3	3	1	1	1	1	1	3	1	3	1	3	3	3	3	3	1	3	1	3	3	3	3	1	3	3	1	1	1	3	3
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	3	3	3	1	3	3	3	3	3	3	3	3	3
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	1	3	3	3	3	3	3	1	3	3	3	3	3	3
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	1	3	3	3	1	3	3	3	3	3
16 PO	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	3	3	1	3	3	3	3	3	1	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	3	3	1	3	3	1	3	3	3	1	3	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	3	3	3	3	3	3	1	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	3	3	3	3	3	3	3	1	3	3	3	3	1	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3	3
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	1	1	3	3	3	3	3	3	3	3
28 Urban	3	1	3	3	1	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	3	1	1	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	3	3	3	3	3	3	1	1	3	1	3	3
31 Opt1	3	1	3	3	3	3	3	1	3	1	1	1	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	3	3	1	1	1	3	1	1	3	3
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	1	1	3	3	1	3	1	1	3	3	3

Amostra 5

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	4	4	1	4	4	1	4	4	1	4	4	4	4	4
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	4	4	4	4	4	1	4	4	4	4	4	4	4
2 Hexse	4	1	1	1	1	4	4	1	4	4	1	4	4	4	1	4	4	1	1	4	4	1	1	1	4	4	4	4	4	4	4
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	1	4	4	4	4	4	1	4	4	4	4
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	4	1	4	4	4	4	4	1	4	4	4
5 CDIII	1	4	4	4	1	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	4	1	4	4	4	4	4	4	1	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Sílv	4	4	4	1	1	4	1	1	1	4	1	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	4	4	4	4	4	4	1	4	4	1	4	4	4	4	1	4	4	4	4	1	4	4	4	4	4	4	1	4	4	4	4
9 Andre	1	4	4	4	4	4	4	4	4	4	4	4	1	4	4	4	4	4	1	4	4	1	4	4	4	4	4	4	4	4	4
10 Estil	4	4	4	4	4	4	4	4	4	4	1	4	1	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	4	4	1	1	1	1	1	4	1	4	1	4	4	4	4	4	1	4	1	4	4	4	1	4	4	1	1	1	1	4	4
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	4	4	4	1	4	4	4	4	4	4	4	4	4	4
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	4	4	4	4	1	4	4	4	4	4	4	4	4
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	1	4	4	4	4	4	1	4	4	4	4	4	4
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	4	1	4	4	4	1	4	4	4	4	4
16 PO	4	1	4	4	4	4	4	1	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	4	4	1	4	4	4	4	4	1	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	4	4	1	4	4	1	4	4	4	1	4	4	1	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	4	4	4	1	4	4	4	4	4	1	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	4	4	4	4	4	4	1	4	4	4	4	1	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	4	4	4	4	4	4	4	1	4	4	4	4	1	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	4	4	4	4	1	4	4	4	4	4	1	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	4	4	4	4	1	1	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	4	1	4	4	4	4	4	1	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 EI	4	4	4	4	4	1	4	4	4	4	4	1	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	4	1	4	4	4	4	4	4	4	4	4	4	4
27 Eil	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	4	1	1	4	4	4	4	4	4	4	4
28 Urban	4	1	4	4	1	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	4	1	1	4	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	4	4	4	4	4	4	4	4	1	1	4	1	4	4
31 Opt1	4	1	4	4	4	4	4	1	4	1	1	1	1	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	1	4	4	1	1	1	4	1	1	4	4
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	4	1	4	1	1	4	4	1	4	1	1	4	4

# Amostra 6

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	1	4	3	1	4	3	1	3	3	4	4	3
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	1	3	3	3	3	1	4	2	4	3	3	3	3	3
2 Hexse	3	1	1	1	1	3	3	1	3	3	1	3	3	3	3	3	1	3	1	3	4	3	1	1	4	3	3	4	3	3	3
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	4	3	1	3	4	3	3	3	3	1	3	3	3
5 CDIII	1	3	3	3	1	3	4	3	4	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	3	1	3	4	4	3	4	3	1	3	4	3	4	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	3	3	3	1	1	3	1	1	1	3	4	4	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	3	3	3	3	3	3	1	3	3	1	3	3	3	3	1	4	3	4	4	1	3	4	4	3	3	1	4	4	3	3	3
9 Andre	1	3	3	3	3	3	3	3	3	3	4	1	3	3	3	5	1	3	3	1	3	3	3	3	3	3	3	4	3	3	3
10 Estll	3	3	4	3	3	3	4	3	3	4	1	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	3	3	1	1	1	1	1	3	1	3	1	4	3	3	4	4	1	3	1	3	3	3	1	4	4	1	1	1	3	3	3
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	4	4	4	1	3	4	3	3	3	3	3	4	4	3	3
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	4	4	4	3	1	4	4	4	3	4	3	3	3	3
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	4	1	4	4	3	4	3	1	3	4	3	4	3	3	3
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	2	4	4	3	4	1	3	3	3	1	4	3	4	3	4	4
16 PO	4	1	3	3	4	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	4	3	3	1	3	3	3	4	1	4	4	3	4	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	3	3	1	3	3	1	4	3	4	1	4	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	4	3	1	3	4	3	3	4	3	1	3	3	3	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	3	4	3	4	4	4	1	4	3	3	3	1	3	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	4	3	4	3	3	3	4	1	4	4	3	3	1	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	3	4	4	3	1	3	4	4	3	4	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	4	4	4	3	1	1	3	3	4	3	4	4	4	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	3	1	4	4	4	4	3	1	3	4	3	4	3	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	3	4	4	3	4	1	3	4	4	4	4	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	4	1	3	3	4	4	3	3	3	3	3	3	3	3
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	4	3	4	4	1	1	4	3	4	4	3	4	3	3	3
28 Urban	3	1	3	4	3	3	3	4	4	3	1	3	3	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	4	1	1	3	3	4	3	4	3	4	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	4	3	4	3	3	4	3	3	1	1	3	1	3	3	3
31 Opt1	3	1	4	4	4	3	3	1	3	1	1	1	1	4	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	4	4	3	1	4	3	1	1	1	3	1	1	3	3	3
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	1	3	1	3	1	1	3	3	1	4	1	1	3	3	3

Amostra 7

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	1	3	3	1	3	3	1	3	3	3	3	3	3
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3
2 Hexse	3	1	1	1	1	3	3	1	3	3	1	3	3	3	1	3	3	1	1	3	3	1	1	1	3	3	3	3	3	3	3
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	3	3	3	3	1	3	3	3	3	3
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	3	3	3	3	3	3	1	3	3	3	3
5 CDIII	1	3	3	3	1	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	3	1	3	3	3	3	3	3	1	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	3	3	3	1	1	3	1	1	1	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	3	3	3	3	3	3	1	3	3	1	3	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
9 Andre	1	3	3	3	3	3	3	3	3	3	3	3	1	3	3	3	3	1	3	3	1	3	3	3	3	3	3	3	3	3	3
10 EstII	3	3	3	3	3	3	3	3	3	3	1	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	3	3	1	1	1	1	1	3	1	3	1	3	3	3	3	3	1	3	1	3	3	3	1	3	3	1	1	1	3	3	3
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	1	3	3	3	3	3	3	3	3	3	3	3
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	3	3	3	1	3	3	3	3	3	3	3	3	3
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	3	3	3	1	3	3	3	3	3	3
16 PO	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	3	3	1	3	3	3	3	3	1	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	3	3	1	3	3	1	3	3	3	1	3	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	3	3	3	3	3	3	1	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	3	3	3	3	3	3	3	1	3	3	3	3	1	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3	3
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3
28 Urban	3	1	3	3	1	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	3	1	1	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	3	3	3	3	3	1	1	3	1	3	3	3
31 Opt1	3	1	3	3	3	3	3	1	3	1	3	1	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	1	3	3	1	1	1	3	1	1	3	3	3
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	1	1	3	3	1	3	1	1	3	3	3

**Amostra 8**

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	1	3	3	1	3	3	1	1	3	3	3	3
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	3	1	3	3	3	3	3	3	3
2 Hexse	3	1	1	1	1	3	3	1	3	3	1	1	1	3	1	3	3	1	1	3	3	1	1	1	3	3	3	3	3	3
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	1	3	3	1	3	3	3	3	3	3	3
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	3	3	3	3	3	3	3	1	3	3	3
5 CDIII	1	3	3	3	1	3	1	3	3	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	3	3	3	3	3	3	3	3	1	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	3	1	3	1	3	3	3	1	1	1	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	3	3	3	3	3	3	1	3	3	1	3	3	3	3	1	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3
9 Andre	3	3	3	3	3	1	3	3	3	3	3	3	1	3	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3	3
10 EstII	3	3	3	3	3	1	3	3	3	3	3	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	3	3	1	1	1	1	1	3	1	3	1	3	3	1	3	3	1	3	3	3	3	3	1	3	3	1	1	1	3	3
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	1	3	3	3	3	3	3	3	3	3	3
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	3	3	3	1	3	3	3	3	3	3	3	3
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	1	3	3	3	3	3	1	3	3	3	3	3	3
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	3	3	3	3	3	1	3	3	3	3	3
16 PO	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	3	3	1	3	3	3	3	3	1	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	3	3	1	3	3	1	3	3	3	1	3	3	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	3	3	3	1	3	3	3	3	3	1	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	3	3	3	3	3	3	1	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	3	3	3	3	3	3	3	1	3	3	3	3	1	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	3	3	3	3	1	3	3	3	3	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	3	1	3	3	3	3	3	1	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	3	3	3	3	3	1	3	3	3	3	3	1	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	3	3	3	3	3	3	3	3	3	3
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	3	1	1	3	3	3	3	3	3	3	3
28 Urban	3	3	3	3	1	1	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	3	3	1	1	3	3	3	3	3	3	1	3	3	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	3	3	3	3	3	3	3	3	1	1	3	1	3	3
31 Opt1	3	1	3	3	3	3	3	1	3	1	1	1	1	3	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	3	3	3	1	3	3	1	1	1	3	1	1	3	3
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	3	1	3	1	3	1	1	3	3	1	3	1	1	3	3

**Amostra 9**

Professor	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0 Elian	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	1	5	5	1	5	5	1	5	5	5	5	5
1 CDI	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5
2 Hexse	5	5	1	5	1	5	5	5	5	5	1	5	5	5	5	5	5	1	1	5	5	1	1	1	5	5	5	5	5	5	5
3 IA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	1	5	5	5	5
4 GA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	1	5	5	5
5 CDIII	1	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6 AL	5	1	5	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
7 Silvi	5	5	5	1	1	5	1	1	1	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
8 Elias	5	5	5	5	5	5	1	5	5	1	5	5	5	5	1	5	5	5	5	1	5	5	5	5	5	5	1	5	5	5	5
9 Andre	1	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	1	5	5	1	5	5	5	5	5	5	5	5	5	5
10 Estll	5	5	5	5	5	5	5	5	5	5	1	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
11 Heinz	5	5	1	1	1	1	1	5	1	5	1	5	5	5	5	5	1	5	1	5	5	5	1	5	5	1	1	1	1	5	5
12 Rosin	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5
13 ITGA	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	1	5	5	5	5	5	5	5	5	5
14 Helio	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5
15 Casti	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	1	5	5	5	5	5	5
16 PO	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
17 Setem	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
18 Marti	5	5	1	5	5	1	5	5	5	1	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
19 Carme	5	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
20 Eleni	5	5	5	5	5	5	1	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
21 Miche	1	5	5	5	5	5	5	5	1	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
22 Wagne	5	5	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23 OB	5	5	5	5	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
24 Razer	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25 El	5	5	5	5	5	1	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
26 Sunye	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	1	5	5	5	5	5	5	5	5	5	5	5	5
27 Ell	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	1	5	5	5	5	5	5	5	5	5
28 Urban	5	1	5	5	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
29 TGII	1	5	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
30 TGIII	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5	5	5	5	5	1	1	5	1	5	5
31 Opt1	5	1	5	5	5	5	5	1	5	1	1	1	1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32 Opt2	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	5	1	1	1	5	1	1	5	5
33 Opt3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	1	5	1	5	1	1	5	5	1	5	1	1	5	5	5

## Relacionamento entre disciplinas e Turmas

Na tabela 10 a seguir está representado o relacionamento das disciplinas com as respectivas turmas, sinalizado com o símbolo X. Nas colunas 1a 8 estão as turmas e nas linhas as disciplinas. As disciplinas sinalizadas com \* referem-se a Estágio Supervisionado, ficam fora da Grade gerada.

Disciplinas	1	2	3	4	5	6	7	8
0 CI066	X							
1 CM201	X							
2 CI063	X							
3 CI055	X							
4 CM046	X							
5 CM045	X							
6 CM202		X						
7 CM005		X						
8 CI067		X						
9 CIA56		X						
10 CE003		X						
11 CI068		X						
12 CI210			X					
13 CI064			X					
14 SA214			X					
15 CI057			X					
16 CI202			X					
17 CI237			X					
18 CI212				X				
19 CM224				X				
20 CI219				X				
21 CI065				X				
22 CI060				X				
23 CI059				X				
24 CI069				X				
25 CI062					X			
26 CI215					X			
27 HB022					X			
28 CI058					X			
29 CI211					X			
30 CI235	*							
31 CI218						X		
32 CI214						X		
33 CI061						X		
34 CI236	*							
35 CI221							X	
36 CI220							X	
37 CI233							X	
38 CI234								X
39 OP206					X	X	X	X
40 OP089					X	X	X	X
41 OP085					X	X	X	X
42 OP209					X	X	X	X
43 OP092					X	X	X	X
44 OP205					X	X	X	X
45 OP204					X	X	X	X
46 OP090					X	X	X	X
47 OP088					X	X	X	X
48 OP090					X	X	X	X
49 OP091					X	X	X	X
50 OP094					X	X	X	X

Tabela 10

## Relacionamento entre Professores e Disciplinas

Na tabela a seguir está representado o relacionamento existente entre disciplinas e professor. As disciplinas (coluna **Discip.**) estão representadas pelo código apresentado na tabela 10.

**Tabela 11**

Professores		Discip.		
0	Elia	0	3	
1	CDI I	1		
2	Hexse	2	12	18
3	IA	4		
4	GA	5		
5	CDIII	6		
6	AL	7		
7	Silvi	8	20	35
8	Elias	9	43	33
9	Andre	21	32	37
10	Estil	10		
11	Heinz	11	16	50
12	Rosin	13		
13	ITGA	14		
14	Helio	15		
15	Castil	17		
16	PO	19		
17	Setem	20		
18	Marti	22	29	
19	Carme	23		
20	Eleni	24		
21	Miche	25		
22	Wagne	26		
23	OB	27		
24	Razer	28		
25	Eli	30		
26	Sunye	31		
27	Eli	34		
28	Urban	36		
29	TGI	37		
30	TGII	38	47	
31	Opt 1	39	42	44
32	Opt 2	40	45	46
33	Opt 3	41	48	49



## **Anexo B**

## Resultados obtidos com as amostras

Abaixo estão apresentadas as soluções ótimas obtidas pelo Algoritmo Cooperativo através da aplicação das amostras de disponibilidade de horários. As soluções estão ordenadas conforme a amostra.

Tabela de horários produzida para amostra 0

PERIODO 1.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI055 /Eliani	CI055 /Eliani	CM045 /GA	CM201 /CDI I	VAGO	/VAGO
19:00	CI066 /Eliani	CI066 /Eliani	CM201 /CDI I	CM046 /IA	VAGO	/VAGO
21:00	CI063 /Hexse	CI063 /Hexse	CM046 /IA	CM046 /GA	VAGO	/VAGO
PERIODO 2.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI068 /Heinz	CI068 /Heinz	CM202 /CDIII	CM005 /AL	VAGO	/VAGO
15:30	CIA56 /Eliani	CI067 /Silvi	CE003 /EstII	CE003 /EstII	VAGO	/VAGO
17:30	CI067 /Silvi	CIA56 /Eliani	CM005 /AL	CM202 /CDIII	VAGO	/VAGO
PERIODO 3.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI202 /Heinz	CI210 /Hexse	CI057 /Helio	CI057 /Helio	VAGO	/VAGO
19:00	CI210 /Hexse	SA214 /ITGA	SA214 /ITGA	CI202 /Heinz	VAGO	/VAGO
21:00	CI237 /Casti	CI064 /Rosin	CI064 /Rosin	CI237 /Casti	VAGO	/VAGO
PERIODO 4.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI219 /Setem	CI065 /Andre	CI219 /Setem	CM224 /PO	CI069 /Eleni	
15:30	CI065 /Andre	CI060 /Marti	CI059 /Carme	CI059 /Carme	VAGO	/VAGO
17:30	CI212 /Hexse	CI212 /Hexse	CM224 /PO	CI069 /Eleni	VAGO	/VAGO
PERIODO 5.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI062 /Miche	CI215 /Wagne	CI211 /Marti	CI215 /Wagne	VAGO	/VAGO
15:30	CI211 /Marti	OP094 /Heinz	OP094 /Heinz	CI062 /Miche	VAGO	/VAGO
17:30	CI058 /Razer	CI058 /Razer	HR022 /OR	VAGO	/VAGO	/VAGO
PERIODO 6.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	OP085 /Opt3	CI061 /Eliani	OP089 /Opt3	CI214 /Andre	VAGO	/VAGO
19:00	CI214 /Andre	OP089 /Opt2	CI218 /Sunye	VAGO	/VAGO	/VAGO
21:00	CI061 /Eliani	CI218 /Sunye	OP085 /Opt3	VAGO	/VAGO	/VAGO
PERIODO 7.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	OP209 /Opt1	OP205 /Opt1	CI233 /TGII	OP209 /Opt1	VAGO	/VAGO
15:30	CI221 /Silvi	CI220 /Urban	CI220 /Urban	VAGO	/VAGO	/VAGO
17:30	OP205 /Opt1	CI233 /TGII	CI221 /Silvi	VAGO	/VAGO	/VAGO
PERIODO 8.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI234 /TGIII	OP090 /Opt3	OP091 /Opt3	CI234 /TGIII	VAGO	/VAGO
19:00	OP088 /TGIII	OP091 /Opt3	OP088 /TGIII	VAGO	/VAGO	/VAGO
21:00	OP090 /Opt3	OP204 /Opt2	OP204 /Opt2	VAGO	/VAGO	/VAGO

# Tabela de horários produzida para amostra 1

PERIODO 1.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI066 /Elian	CI055 /Elian	CM201 /CDI I	CI066 /Elian	VAGO /VAGO	
19:00	CI055 /Elian	CM045 /GA	CM045 /GA	CM046 /IA	VAGO /VAGO	
21:00	CI063 /Hexse	CI063 /Hexse	CM046 /IA	CM201 /CDI I	VAGO /VAGO	
PERIODO 2.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI068 /Heinz	CI067 /Silvi	CE003 /EstII	CIA56 /Elias	VAGO /VAGO	
15:30	CI067 /Silvi	CI068 /Heinz	CM202 /CDIII	CM005 /AL	VAGO /VAGO	
17:30	CM202 /CDIII	CIA56 /Elias	CM005 /AL	CE003 /EstII	VAGO /VAGO	
PERIODO 3.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI202 /Heinz	CI202 /Heinz	SA214 /ITGA	CI237 /Casti	VAGO /VAGO	
19:00	CI210 /Hexse	CI064 /Rosin	CI057 /Helio	SA214 /ITGA	VAGO /VAGO	
21:00	CI237 /Casti	CI057 /Helio	CI210 /Hexse	CI064 /Rosin	VAGO /VAGO	
PERIODO 4.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI065 /Andre	CI212 /Hexse	CI065 /Andre	CI059 /Carme	CI069 /Eleni	
15:30	CI219 /Setem	CI060 /Marti	CM224 /PO	CM224 /PO	VAGO /VAGO	
17:30	CI212 /Hexse	CI219 /Setem	CI059 /Carme	CI069 /Eleni	VAGO /VAGO	
PERIODO 5.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI215 /Wagne	CI058 /Razer	OP094 /Heinz	CI211 /Marti	VAGO /VAGO	
15:30	CI211 /Marti	CI215 /Wagne	CI058 /Razer	CI062 /Miche	VAGO /VAGO	
17:30	HB022 /OB	OP094 /Heinz	CI062 /Miche	VAGO /VAGO	VAGO /VAGO	
PERIODO 6.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI061 /Elias	CI214 /Andre	CI218 /Sunye	OP085 /Opt3	VAGO /VAGO	
19:00	VAGO /VAGO	CI061 /Elias	OP085 /Opt3	OP089 /Opt2	VAGO /VAGO	
21:00	CI214 /Andre	CI218 /Sunye	OP089 /Opt2	VAGO /VAGO	VAGO /VAGO	
PERIODO 7.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI221 /Silvi	CI233 /TGII	CI220 /Urban	CI221 /Silvi	VAGO /VAGO	
15:30	OP209 /Opt1	CI220 /Urban	OP205 /Opt1	VAGO /VAGO	VAGO /VAGO	
17:30	CI233 /TGII	OP205 /Opt1	OP209 /Opt1	VAGO /VAGO	VAGO /VAGO	
PERIODO 8.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	OP088 /TGIII	OP204 /Opt2	OP204 /Opt2	CI234 /TGIII	VAGO /VAGO	
19:00	CI234 /TGIII	OP091 /Opt3	OP088 /TGIII	VAGO /VAGO	VAGO /VAGO	
21:00	OP090 /Opt3	OP090 /Opt3	OP091 /Opt3	VAGO /VAGO	VAGO /VAGO	

# Tabela de horários produzida para amostra 2

PERIODO 1.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI066 /Eliani	VAGO /VAGO	CM046 /IA	CI055 /Eliani	CI066 /Eliani	
19:00	CI066 /Eliani	VAGO /VAGO	CI063 /Hexse	CM046 /IA	CM046 /GA	
21:00	CI063 /Hexse	CM201 /CDI II	CM201 /CDI II	CM046 /GA	VAGO /VAGO	
PERIODO 2.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CE003 /EstIII	CI067 /Silvi	CIA56 /Elias	CI067 /Silvi	VAGO /VAGO	
15:30	CM005 /AL	CI068 /Heinz	CM202 /CDIII	CE003 /EstIII	VAGO /VAGO	
17:30	CI068 /Heinz	CIA56 /Elias	CM005 /AL	CM202 /CDIII	VAGO /VAGO	
PERIODO 3.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI210 /Hexse	CI210 /Hexse	SA214 /ITGA	CI057 /Helio	VAGO /VAGO	
19:00	CI202 /Heinz	CI064 /Rosin	CI057 /Helio	CI064 /Rosin	VAGO /VAGO	
21:00	SA214 /ITGA	CI237 /Casti	CI202 /Heinz	CI237 /Casti	VAGO /VAGO	
PERIODO 4.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI065 /Andre	CI069 /Eleni	CI059 /Carmel	CI059 /Carmel	CI065 /Andre	
15:30	CM224 /PO	CI219 /Setem	CM224 /PO	CI060 /Marti	VAGO /VAGO	
17:30	CI212 /Hexse	CI212 /Hexse	CI069 /Eleni	CI219 /Setem	VAGO /VAGO	
PERIODO 5.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI062 /Miche	CI058 /Razer	HB022 /OB	CI211 /Marti	VAGO /VAGO	
15:30	CI215 /Wagne	CI211 /Marti	CI062 /Miche	CI215 /Wagne	VAGO /VAGO	
17:30	CI058 /Razer	OP094 /Heinz	OP094 /Heinz	VAGO /VAGO	VAGO /VAGO	
PERIODO 6.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	CI061 /Elias	VAGO /VAGO	CI218 /Sunye	OP085 /Opt3	VAGO /VAGO	
19:00	CI218 /Sunye	OP089 /Opt2	OP085 /Opt3	CI061 /Elias	VAGO /VAGO	
21:00	CI214 /Andre	CI214 /Andre	OP089 /Opt2	VAGO /VAGO	VAGO /VAGO	
PERIODO 7.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
13:30	CI233 /TGII	VAGO /VAGO	OP205 /Opt1	OP209 /Opt1	CI233 /TGII	
15:30	OP205 /Opt1	CI220 /Urban	CI220 /Urban	CI221 /Silvi	VAGO /VAGO	
17:30	OP209 /Opt1	CI221 /Silvi	VAGO /VAGO	VAGO /VAGO	VAGO /VAGO	
PERIODO 8.						
-HOR-----	Segunda-----	Terça-----	Quarta-----	Quinta-----	Sexta-----	
17:30	VAGO /VAGO	OP091 /Opt3	OP090 /Opt3	OP088 /TGIII	VAGO /VAGO	
19:00	CI234 /TGIII	OP090 /Opt3	OP204 /Opt2	CI234 /TGIII	VAGO /VAGO	
21:00	OP091 /Opt3	OP088 /TGIII	VAGO /VAGO	OP204 /Opt2	VAGO /VAGO	

# Tabela de horários produzida para amostra 3

## PERIODO 1.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
17:30	CI066 /Eliani	CI066 /Eliani	CI055 /Eliani	CI055 /Eliani	VAGO /VAGO
19:00	CM201 /CDI I	CM046 /IA	CM201 /CDI I	CM046 /IA	VAGO /VAGO
21:00	CI063 /Hexse	CM045 /GA	CI063 /Hexse	CM045 /GA	VAGO /VAGO

## PERIODO 2.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CM202 /CDIII	CI067 /Silvi	CIA56 /Eliani	CIA56 /Eliani	CE003 /EstII
15:30	CM005 /AL	CM202 /CDIII	CI067 /Silvi	CE003 /EstIII	VAGO /VAGO
17:30	CI068 /Heinz	CI068 /Heinz	CM005 /AL	VAGO /VAGO	VAGO /VAGO

## PERIODO 3.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
17:30	CI064 /Rosini	CI210 /Hexse	SA214 /ITGA	CI237 /Casti	CI202 /Heinz
19:00	SA214 /ITGA	CI064 /Rosini	CI210 /Hexse	CI202 /Heinz	VAGO /VAGO
21:00	CI057 /Helio	CI237 /Casti	CI057 /Helio	VAGO /VAGO	VAGO /VAGO

## PERIODO 4.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CI065 /Andre	CI059 /Carme	CI069 /Eleni	CI059 /Carme	CI065 /Andre
15:30	CM224 /PO	CI212 /Hexse	CM224 /PO	CI212 /Hexse	VAGO /VAGO
17:30	CI219 /Setem	CI060 /Marti	CI219 /Setem	CI069 /Eleni	VAGO /VAGO

## PERIODO 5.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CI062 /Miche	OP094 /Heinz	OP094 /Heinz	CI211 /Marti	VAGO /VAGO
15:30	CI058 /Razer	CI215 /Wagne	CI058 /Razer	CI215 /Wagne	VAGO /VAGO
17:30	CI211 /Marti	HB022 /OB	CI062 /Miche	VAGO /VAGO	VAGO /VAGO

## PERIODO 6.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
17:30	VAGO /VAGO	CI219 /Sunye	OP085 /Opt3	OP085 /Opt3	OP089 /Opt2
19:00	CI218 /Sunye	CI061 /Eliani	OP089 /Opt2	CI061 /Eliani	VAGO /VAGO
21:00	CI214 /Andre	CI214 /Andre	VAGO /VAGO	VAGO /VAGO	VAGO /VAGO

## PERIODO 7.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
13:30	VAGO /VAGO	CI233 /TGII	VAGO /VAGO	OP205 /Opt1	OP205 /Opt1
15:30	CI220 /Urban	CI220 /Urban	OP209 /Opt1	CI221 /Silvi	VAGO /VAGO
17:30	CI233 /TGII	VAGO /VAGO	CI221 /Silvi	OP209 /Opt1	VAGO /VAGO

## PERIODO 8.

-HOR-	Segunda	Terça	Quarta	Quinta	Sexta
17:30	CI234 /TGIII	OP091 /Opt3	VAGO /VAGO	OP088 /TGIII	OP088 /TGIII
19:00	OP091 /Opt3	OP204 /Opt2	VAGO /VAGO	CI234 /TGIII	VAGO /VAGO
21:00	VAGO /VAGO	OP090 /Opt3	OP204 /Opt2	OP090 /Opt3	VAGO /VAGO

# Tabela de horários produzida para amostra 4

## PERIODO 1.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
17:30	CI055 /Elian	CI055 /Elian	CI066 /Elian	CI066 /Elian	VAGO /VAGO
19:00	CM201 /CDI I	CM046 /IA	CM201 /CDI I	CM046 /IA	VAGO /VAGO
21:00	CI063 /Hexse	CM045 /GA	CI063 /Hexse	CM045 /GA	VAGO /VAGO

## PERIODO 2.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CM202 /CDIII	CI067 /Silvi	CIA56 /Elias	CIA56 /Elias	CE003 /EstII
15:30	CM005 /AL	CM202 /CDIII	CI067 /Silvi	CE003 /EstIII	VAGO /VAGO
17:30	CI068 /Heinz	CI068 /Heinz	CM005 /AL	VAGO /VAGO	VAGO /VAGO

## PERIODO 3.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
17:30	CI064 /Rosin	CI210 /Hexse	SA214 /ITGA	CI237 /Casti	CI202 /Heinz
19:00	SA214 /ITGA	CI064 /Rosin	CI210 /Hexse	CI202 /Heinz	VAGO /VAGO
21:00	CI057 /Helio	CI237 /Casti	CI057 /Helio	VAGO /VAGO	VAGO /VAGO

## PERIODO 4.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CI065 /Andre	CI059 /Carme	CI069 /Eleni	CI059 /Carme	CI065 /Andre
15:30	CM224 /PO	CI212 /Hexse	CM224 /PO	CI212 /Hexse	VAGO /VAGO
17:30	CI219 /Setem	CI060 /Marti	CI219 /Setem	CI069 /Eleni	VAGO /VAGO

## PERIODO 5.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
13:30	CI062 /Miche	OP094 /Heinz	OP094 /Heinz	CI211 /Marti	VAGO /VAGO
15:30	CI058 /Razer	CI215 /Wagne	CI058 /Razer	CI215 /Wagne	VAGO /VAGO
17:30	CI211 /Marti	HB022 /OB	CI062 /Miche	VAGO /VAGO	VAGO /VAGO

## PERIODO 6.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
17:30	VAGO /VAGO	CI218 /Sunye	VAGO /VAGO	OP089 /Opt2	OP085 /Opt3
19:00	CI218 /Sunye	CI061 /Elias	VAGO /VAGO	CI061 /Elias	VAGO /VAGO
21:00	CI214 /Andre	CI214 /Andre	OP089 /Opt2	OP085 /Opt3	VAGO /VAGO

## PERIODO 7.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
13:30	VAGO /VAGO	CI233 /TGII	CI221 /Silvi	OP209 /Opt1	OP209 /Opt1
15:30	CI220 /Urban	CI220 /Urban	OP205 /Opt1	CI221 /Silvi	VAGO /VAGO
17:30	CI233 /TGII	VAGO /VAGO	VAGO /VAGO	OP205 /Opt1	VAGO /VAGO

## PERIODO 8.

HOR	Segunda	Terça	Quarta	Quinta	Sexta
17:30	OP088 /TGIII	OP091 /Opt3	OP091 /Opt3	OP088 /TGIII	CI234 /TGIII
19:00	OP090 /Opt3	OP204 /Opt2	OP204 /Opt2	CI234 /TGIII	VAGO /VAGO
21:00	VAGO /VAGO	OP090 /Opt3	VAGO /VAGO	VAGO /VAGO	VAGO /VAGO

Tabela de horários produzida para amostra 5

PERIODO 1.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI066 /Elian	CI066 /Elian	CI055 /Elian	CI055 /Elian	VAGO /VAGO	
19:00	CM201 /CDI I	CM046 /IA	CM201 /CDI I	CM046 /IA	VAGO /VAGO	
21:00	CI063 /Hexse	CM045 /GA	CI063 /Hexse	CM045 /GA	VAGO /VAGO	
PERIODO 2.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CM202 /CDIII	CI067 /Silvi	CIA56 /Elias	CIA56 /Elias	CE003 /EstII	
15:30	CM005 /AL	CM202 /CDIII	CI067 /Silvi	CE003 /EstII	VAGO /VAGO	
17:30	CI068 /Heinz	CI068 /Heinz	CM005 /AL	VAGO /VAGO	VAGO /VAGO	
PERIODO 3.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI064 /Rosin	CI210 /Hexse	SA214 /ITGA	CI237 /Casti	CI202 /Heinz	
19:00	SA214 /ITGA	CI064 /Rosin	CI210 /Hexse	CI202 /Heinz	VAGO /VAGO	
21:00	CI057 /Helio	CI237 /Casti	CI057 /Helio	VAGO /VAGO	VAGO /VAGO	
PERIODO 4.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CI065 /Andre	CI059 /Carme	CI069 /Eleni	CI059 /Carme	CI065 /Andre	
15:30	CM224 /PO	CI212 /Hexse	CM224 /PO	CI212 /Hexse	VAGO /VAGO	
17:30	CI219 /Setem	CI060 /Martí	CI219 /Setem	CI069 /Eleni	VAGO /VAGO	
PERIODO 5.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CI211 /Martí	CI058 /Razer	CI215 /Wagne	OP094 /Heinz	VAGO /VAGO	
15:30	CI211 /Martí	CI058 /Razer	OP094 /Heinz	CI215 /Wagne	VAGO /VAGO	
17:30	CI062 /Miche	CI062 /Miche	HB022 /OB	VAGO /VAGO	VAGO /VAGO	
PERIODO 6.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	VAGO /VAGO	CI218 /Sunye	VAGO /VAGO	OP089 /Opt2	OP085 /Opt3	
19:00	CI218 /Sunye	CI061 /Elias	VAGO /VAGO	CI061 /Elias	VAGO /VAGO	
21:00	CI214 /Andre	CI214 /Andre	OP089 /Opt2	OP085 /Opt3	VAGO /VAGO	
PERIODO 7.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	VAGO /VAGO	CI233 /TGII	CI221 /Silvi	OP205 /Opt1	OP209 /Opt1	
15:30	CI220 /Urban	CI220 /Urban	OP205 /Opt1	CI221 /Silvi	VAGO /VAGO	
17:30	CI233 /TGII	VAGO /VAGO	VAGO /VAGO	OP209 /Opt1	VAGO /VAGO	
PERIODO 8.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI234 /TGIII	OP091 /Opt3	OP090 /Opt3	CI234 /TGIII	OP088 /TGIII	
19:00	OP091 /Opt3	OP204 /Opt2	OP204 /Opt2	OP088 /TGIII	VAGO /VAGO	
21:00	VAGO /VAGO	OP090 /Opt3	VAGO /VAGO	VAGO /VAGO	VAGO /VAGO	

Tabela de horários produzida para amostra 6

PERÍODO 1.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI055 /Elian	CI055 /Elian	CI066 /Elian	CI066 /Elian	VAGO /VAGO	
19:00	CI063 /Hexse	CM046 /IA	CM201 /CDI II	CM046 /IA	VAGO /VAGO	
21:00	CM201 /CDI II	CM045 /GA	CI063 /Hexse	CM045 /GA	VAGO /VAGO	
PERÍODO 2.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CM202 /CDI III	CI067 /Silvi	CIA56 /Elias	CIA56 /Elias	CE003 /EstII	
15:30	CM005 /AL	CM202 /CDI III	CI067 /Silvi	CE003 /EstII	VAGO /VAGO	
17:30	CI068 /Heinz	CI068 /Heinz	CM005 /AL	VAGO /VAGO	VAGO /VAGO	
PERÍODO 3.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI064 /Rosini	CI210 /Hexse	SA214 /ITGA	CI237 /Casti	CI202 /Heinz	
19:00	SA214 /ITGA	CI064 /Rosini	CI210 /Hexse	CI202 /Heinz	VAGO /VAGO	
21:00	CI057 /Helio	CI237 /Casti	CI057 /Helio	VAGO /VAGO	VAGO /VAGO	
PERÍODO 4.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CI065 /Andre	CI219 /Setem	CI069 /Eleni	CI059 /Carme	CI065 /Andre	
15:30	CM224 /PO	CI212 /Hexse	CM224 /PO	CI212 /Hexse	VAGO /VAGO	
17:30	CI059 /Carme	CI060 /Martii	CI219 /Setem	CI069 /Eleni	VAGO /VAGO	
PERÍODO 5.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	CI211 /Martii	CI062 /Michel	CI062 /Michel	OP094 /Heinz	VAGO /VAGO	
15:30	CI211 /Martii	CI058 /Razer	CI215 /Wagne	CI058 /Razer	VAGO /VAGO	
17:30	CI215 /Wagne	HB022 /OB	OP094 /Heinz	VAGO /VAGO	VAGO /VAGO	
PERÍODO 6.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	VAGO /VAGO	CI218 /Sunye	OP085 /Opt3	OP089 /Opt2	OP085 /Opt3	
19:00	CI218 /Sunye	CI061 /Elias	OP089 /Opt2	CI061 /Elias	VAGO /VAGO	
21:00	CI214 /Andre	CI214 /Andre	VAGO /VAGO	VAGO /VAGO	VAGO /VAGO	
PERÍODO 7.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
13:30	VAGO /VAGO	CI233 /TGII	VAGO /VAGO	OP209 /Opt1	OP205 /Opt1	
15:30	CI220 /Urban	CI221 /Silvi	OP209 /Opt1	CI220 /Urban	VAGO /VAGO	
17:30	CI233 /TGII	VAGO /VAGO	CI221 /Silvi	OP205 /Opt1	VAGO /VAGO	
PERÍODO 8.						
-HOR-	Segunda	Terça	Quarta	Quinta	Sexta	
17:30	CI234 /TGIII	OP090 /Opt3	VAGO /VAGO	OP088 /TGIII	OP088 /TGIII	
19:00	OP090 /Opt3	OP204 /Opt2	VAGO /VAGO	CI234 /TGIII	VAGO /VAGO	
21:00	VAGO /VAGO	OP091 /Opt3	OP204 /Opt2	OP091 /Opt3	VAGO /VAGO	